On Transforming Reinforcement Learning With Transformers: The Development Trajectory

Shengchao Hu, Li Shen, Ya Zhang, Yixin Chen, *Fellow, IEEE*, and Dacheng Tao, *Fellow, IEEE*

(Survey Paper)

***Abstract*—Transformers, originally devised for natural language processing (NLP), have also produced significant successes in computer vision (CV). Due to their strong expression power, researchers are investigating ways to deploy transformers for reinforcement learning (RL), and transformer-based models have manifested their potential in representative RL benchmarks. In this paper, we collect and dissect recent advances concerning the transformation of RL with transformers (transformer-based RL (TRL)) to explore the development trajectory and future trends of this field. We group the existing developments into two categories: architecture enhancements and trajectory optimizations, and examine the main applications of TRL in robotic manipulation, text-based games (TBGs), navigation, and autonomous driving. Architecture enhancement methods consider how to apply the powerful transformer structure to RL problems under the traditional RL framework, facilitating more precise modeling of agents and environments compared to traditional deep RL techniques. However, these methods are still limited by the inherent defects of traditional RL algorithms, such as bootstrapping and the "deadly triad". Trajectory optimization methods treat RL problems as sequence modeling problems and train a joint state-action model over entire trajectories under the behavior cloning framework; such approaches are able to extract policies from static datasets and fully use the long-sequence modeling capabilities of transformers. Given these advancements, the limitations and challenges in TRL are reviewed and proposals regarding future research directions are discussed. We hope that this survey can provide a detailed introduction to TRL and motivate future research in this rapidly developing field.**

***Index Terms*—Literature survey, reinforcement learning, representation learning, transformer.**

I. Introduction

RECENTLY, the transformer architecture has made substantial progress in natural language processing (NLP) tasks [1]. For example, generative pretraining (GPT) series models [2] and bidirectional encoder representations from transformers (BERT) models [3] have achieved state-of-the-art performance on a wide range of downstream tasks (e.g. question answering (QA) and sentence classification). Inspired by the success of the transformer architecture in NLP, researchers have also tried to apply transformers to computer vision (CV) tasks. Chen et al. [4] utilized a transformer to auto-regressively predict pixels, and a vision transformer (ViT) [5] was directly applied to sequences of image patches to classify full images. This approach has achieved state-of-the-art performance on multiple image recognition benchmarks. Considering an image as a sequence of sub-images [5], [6], the transformer architecture is able to extract representative features via its self-attention mechanism [7], [8] and can be further used to generate more powerful multi-modal visual-language models, such as DALL-E [9], Flamingo [10], and Gato [11].

Reinforcement learning [12] (RL) is a powerful control strategy that usually consists of an environment and an agent. The agent observes the current state of the environment, makes actions, and obtains the reward for the current action and the state of the next moment; the goal of the agent is to maximize the cumulative reward it obtains. The advent of deep neural networks has catalyzed the popularity of deep RL, leveraging their extensive capacity to make function approximation methods more accurate and enable agents to operate normally in unstructured environments. However, current deep RL approaches mainly rely on interacting with the environment to dynamically collect data, constraining the amount of training data collected. This becomes particularly challenging in domains where interactions are costly, such as robotic manipulation [13], education [14], healthcare [15], and autonomous driving [16]. Thus, offline RL [17], [18] has attracted researchers' interest as a data-driven learning method that can extract policies from static previously collected datasets without interacting with the environment. The offline RL setting can make full use of the ability of deep networks to extract the optimal policy from a large amount of offline data, but it also needs to address the distribution discrepancy between the offline training data and the target policy.

Due to the sequential decision process of RL, a natural idea is to apply transformers, which have been fully developed in recent years [19], [20], to augment deep RL methods. Initially, transformers were typically used as alternative architectures for replacing convolutional neural networks (CNNs) or long short-term memory (LSTM) in deep RL methods, and they mainly provided memory information for the agent network in environments where the critical observations often spanned the

Manuscript received 21 January 2023; revised 9 May 2024; accepted 29 May 2024. Date of publication 3 June 2024; date of current version 5 November 2024. The work of Li Shen was supported by STI 2030–Major Projects under Grant 2021ZD0201405. The work of Dacheng Tao was supported by NTU RSR and Start Up Grants. Recommended for acceptance by M. Sugiyama. *(Corresponding author: Li Shen.)*

Shengchao Hu and Ya Zhang are with Shanghai Jiao Tong University, Shanghai 200240, China, and also with Shanghai AI Lab, Shanghai 200232, China (e-mail: charles-hu@sjtu.edu.cn; ya_zhang@sjtu.edu.cn).

Li Shen is with Sun Yat-sen University, Guangzhou 510275, China, and also with JD Explore Academy, Beijing 101111, China (e-mail: mathshenli@gmail.com).

Yixin Chen is with Washington University in St Louis, St. Louis, MO 63130 USA (e-mail: chen@cse.wustl.edu).

Dacheng Tao is with Nanyang Technological University, Singapore 639798 (e-mail: dacheng.tao@ntu.edu.sg).

Digital Object Identifier 10.1109/TPAMI.2024.3408271

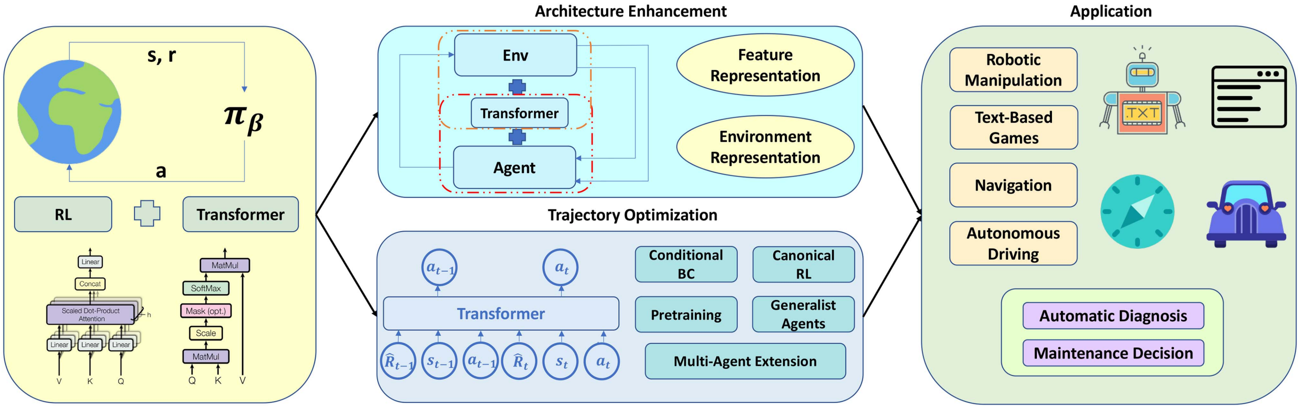

Fig. 1. A detailed overview of the transformers' involvement in RL and realistic application. The Architecture Enhancement block demonstrates the specific part of transformer participation in traditional RL methods. The Trajectory Optimization block shows the usage of the transformer under the sequence modeling framework and corresponding research categories. Finally, the Application block introduces several realistic applications of TRL.

entire episode. However, these methods suffer from the instability issue from an empirical standpoint when directly applying the original transformer structure to the decision process [21]; thus, GtrXL [19] first focused on modifying the transformer architecture to better adapt it to RL, and it has motivated follow-up works that have provided more powerful transformer architectures [22], [23]. On the other hand, the decision transformer (DT) [24] and trajectory transformer (TT) [25] redefine the RL challenge as a conditional sequence modeling task, exploiting the transformer's proficiency in modeling long sequences. This approach has spurred a wave of research, expanding the capabilities of transformer-based RL (TRL) Despite these advancements, there remains a discernible gap in a cohesive synthesis and understanding of TRL methodologies. A comprehensive survey of the field would be instrumental in aligning ongoing research efforts with the latest developments, thereby catalyzing further progress in TRL and providing a valuable asset for the research community.

In this paper, we focus on providing a comprehensive overview of recent advances in TRL and discussing the challenges and potential directions for future improvement (shown in Fig. 1). To facilitate future research on different topics, we categorize the developed TRL approaches from the perspective of their methods and applications, as listed in Table I, which include Architecture Enhancements, Trajectory Optimizations, and Applications.

Regarding Architecture Enhancement, we mainly consider how to apply a more powerful transformer structure to RL problems under the traditional RL framework. According to the specific part of the process in which transformers participate, we further divide these methods into Feature Representation and Environmental Representation strategies. Feature Representation approaches mainly include methods where a transformer is used to extract feature representations from multi-modal inputs, and then an agent utilizes these representations to make decisions through value- or policy-based methods. Environmental Representation mainly illustrates how to leverage the transformer architecture for dynamics and reward modeling, which can be used to form better decision processes with additional planning algorithms.

Trajectory Optimization methods all regard RL as a sequence modeling problem and use decision transformers to learn policies under the behavior cloning (BC) framework. Depending on their underlying motivations and intended tasks, these methods can be categorized into five distinct approaches: Conditional BC, Canonical RL, Pretraining, Generalist Agents, and Multi-Agent Extensions. 1) Conditional BC interprets RL as conditional behavior cloning, utilizing the transformer architecture to model the state-action-reward sequence, encompassing pioneering works and their optimality conditions. 2) Canonical RL seeks to amalgamate the benefits of sequence modeling with traditional RL algorithms to enhance original algorithm performance. 3) Pretraining emphasizes refining decision transformers to improve performance in downstream RL tasks, focusing on pretraining datasets and objectives. 4) Generalist Agents are defined by algorithms that empower a single agent to perform adeptly across various tasks and domains. 5) Multi-Agent Extension explore the integration of multi-agent RL (MARL) with sequence modeling, aiming to unlock the potential of modern sequence models for MARL. This paper also highlights four significant applications of TRL: Robotic Manipulation, Text-Based Games (TBGs), Vision-Language Navigation, and Autonomous Driving, demonstrating their relevance and solvability through TRL techniques. Finally, we discuss several challenges and provide insights into the future prospects of this line of research.

The rest of the paper is organized as follows. Section II introduces the preliminaries of this survey, including a brief overview of RL and the foundation of the standard transformer. Sections III and IV are the main parts of the paper, in which we summarize the existing TRL methods and their applications, respectively. Then, in Section V, we briefly discuss several challenges and future directions related to TRL. Finally, we provide a summary to conclude this work. Note that in this survey, we mainly include the representative works, and some recent preprint works on arXiv may be missed due to the rapid development of this field. Complementing our discussion, other surveys like Agarwal et al. [26] and Li et al. [27] examine transformers in RL from different angles, addressing challenges, applications, and offering a taxonomy of their uses in RL domains. Our survey distinguishes itself by focusing on architectural and trajectory optimizations within the broader context of transformer applications in RL, aiming to present a detailed and forward-looking analysis of the field.

TABLE I
Representative Works of TRL

Category	Sub-category	Method	Highlights	Publication
Architecture Enhancement	Feature Representation	WMG [28]	Transformer-based RL with factored observations	ICML 2020
		GTrXL [19]	Replace LSTM in V-MPO with Gated Transformer-XL	ICML 2020
		RFWPs [22]	Add recurrence to the FWP	NeurIPS 2021
		COBERL [29]	Combine contrastive loss and a hybird LSTM-transformer	ICLR 2022
	Environment Representation	TransDreamer [30]	Transformer-based stochastic world model	arXiv 2022
		PlaTe [31]	Learn state and action spaces from videos	RAL 2022
		IRIS [32]	World model composed of a autoencoder and a transformer	ICLR 2023
		MINECLIP [33]	An automatic evaluation metric	NeurIPS 2022
Trajectory Optimization	Conditioned BC	DT [24]	Return-to-go, BC conditioning desired RTG	NeurIPS 2021
		TT [25]	Modeling distribution of trajectory, beam search	NeurIPS 2021
		ESPER [34]	Cluster trajectories, average cluster returns	NeurIPS 2022
		RCSL [35]	Theoretical analysis for capabilities and limitations	NeurIPS 2022
	Canonical RL	ODT [36]	Blends offline pretraining with online finetuning	ICML 2022
		StARformer [37]	Markovian-like inductive bias	ECCV 2022
		QDT [38]	Utilize the Q-function to relabel the RTG	ICML 2023
		CGDT [39]	Align the target returns with the expected returns	AAAI 2024
	Pretraining	ChibiT [40]	Pretrain on the Wikipedia	arXiv 2022
		LID [41]	Pretrained LMs as a general scaffold	NeruIPS 2022
		MaskDP [42]	A pretraining method to learn generalizable models	NeurIPS 2022
		MTM [43]	Pretraining with a highly randomized masking pattern	ICML 2023
	Generalist Agents	Multi-Game DT [44]	Power-law performance trend	NeuIPS 2022
		Gato [11]	Multi-modal, multi-task, multi-embodiment model	TMLR 2022
		GDT [45]	Hindsight information matching generalization	ICLR 2022
		Prompt-DT [46]	Trajectory prompt for generalization	ICML 2022
	Multi-Agent Extension	MADT [47]	Decision Transformer in offline MARL	arXiv 2021
		MAT [48]	Build the bridge between MARL and SMs	NeurIPS 2022
		CommFormer [49]	Transformer for multi-agent communication	ICLR 2024
		MaskMA [50]	Mask-based decision transformer for zero-shot MARL	arXiv 2024
Application	Robotic Manipulation	T-OSIL [51]	Transformer for OSIL	CoRL 2021
		LATTE [52]	Modify the trajectory based on the multi-modal transformer	ICRA 2023
		TTP [53]	Prompt-situation Transformer, multi-preference learning	CoRL 2023
	Text-based Games	Q*BERT [54]	Build a knowledge graph by answering questions	arXiv 2020
		GATA [55]	Learn to construct and update graph-structured beliefs	NeurIPS 2020
		OOTD [56]	Model-based methods for TBGs	ICLR 2022
	Navigation	PREVALENT [57]	Pretraining with finetuning for VLN tasks	CVPR 2021
		VLNBERT [58]	Equip BERT model with recurrent function	CVPR 2021
		HAMT [59]	Hierarchical transformer encoding long-range history	NeurIPS 2021
	Autonomous Driving	TransFuser [60]	Fusion of intermediate features of the front view and LiDAR	CVPR 2021
		InterFuse [61]	Safety-enhanced framework with multi-modal, view sensors	CoRL 2022
		SPLT [62]	Disentangling the policy and world models	ICML 2022

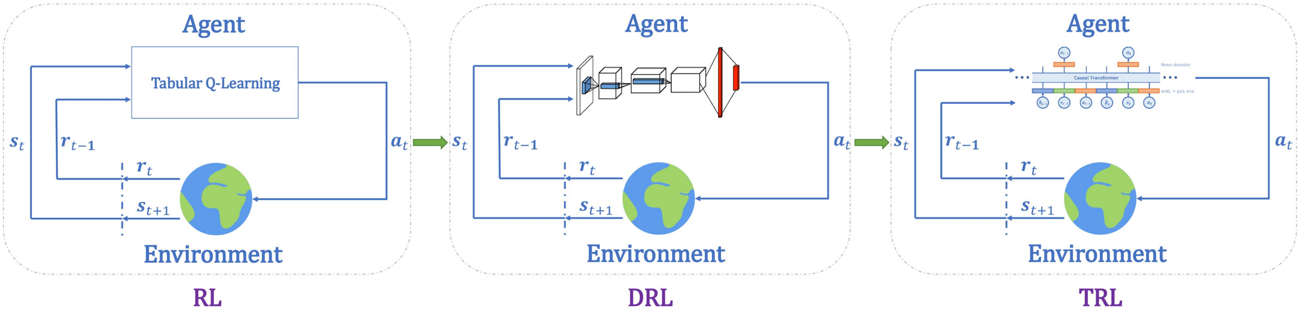

Fig. 2. The development trajectory of RL. At first, the RL agents use the tabular Q-learning to make decisions at the current moment, then the DRL agents begin to use the deep network to estimate the value and policy functions, and finally, the TRL agents leverage the ability of transformer architecture to evaluate the policy. There is a similar trend for environment modeling in the model-based RL.

II. Preliminaries

In this section, we delineate the foundational concepts underpinning this survey.

A. Reinforcement Learning

RL entails the process of iteratively selecting optimal actions based on current states at each timestep to maximize a numerical reward signal, where rewards are allocated to state-action pairs in alignment with the defined problem criteria. Initially, we elucidate the foundational model of a markov decision process (MDP) and highlight its critical attributes. Subsequently, we expound upon the categories of RL, enumerating the commonly encountered components and definitions integral to standard RL problems.

1) MDPs: As a mathematical model of a sequential decision problem, an MDP is considered the ideal modeling approach in dynamic environments and provides the theoretical basis for RL. To be specific, as shown in Fig. 2, an agent is asked to make a decision $a_t \in \mathcal{A}$ based on the current state $s_t \in \mathcal{S}$; then, the environment responds to the action made by the agent and transforms the state to the next state $s_{t+1} \in \mathcal{S}$ with the reward $r_t \in \mathbb{R}$.

Formally, an MDP can be defined by a 6-tuple notation $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{T}(s_{t+1}|s_t, a_t)$ denotes the transition probability from state s_t to s_{t+1} after taking action a_t, ρ_0 denotes the initial state distribution, $\mathcal{R}(s_t, a_t)$ denotes the reward value after taking action a_t at state s_t and $\gamma \in (0, 1]$ denotes the discount factor. Under the MDP setting, a trajectory is defined as the experience of the agent, which is denoted by $\tau = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_t, a_t, r_t)$, and a policy is defined as a probability function $\pi(a_t|s_t)$, which represents the probability of taking action a_t at state s_t. Our objective is to identify the optimal policy $\pi^*(a|s)$, which maximizes the expected cumulative reward over all possible trajectories generated by the policy, as formally expressed:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=1}^{T} r_t\right]. \quad (1)$$

2) Categories of RL: RL encompasses a spectrum of algorithms, each presenting unique benefits and limitations [63]. In the subsequent part, we categorize various RL algorithms, emphasizing those most pertinent to the integration and advancements of transformers in the RL domain.

Model-Based & Model-Free RL: In model-based RL [64], a transition function is learned using trajectories τ generated by environmental interaction. This function estimates the probability distribution $p(s_{t+1}, r_t|s_t, a_t)$ for predicting subsequent states s_{t+1} and rewards r_t, based on current state s_t and action a_t. Utilizing this model, agents can plan and execute actions to optimize the expected cumulative reward $\sum_{t=1}^{T} r_t$, which boosts the sample efficiency by using relatively few interactions. Conversely, model-free RL strategies derive optimal actions through direct environmental engagement, without modeling state-transition dynamics. This approach often leads to slower convergence and reduced sample efficiency compared to model-based RL [65]. Nonetheless, the adaptability of model-free RL to environmental variations enhances its robustness in complex or stochastic settings [66], [67], [68]. Additionally, model-free RL is computationally less demanding due to its elimination of environmental model learning.

On-Policy & Off-Policy RL: On-policy RL [12] approaches employ the current policy for collecting transitions to update the value function. Despite their straightforward implementation, on-policy methods exhibit several limitations, including sample inefficiency [69], which necessitates extensive environmental interaction for optimal performance. Furthermore, they are prone to policy oscillation and instability [70], impeding exploration and leading to potentially sub-optimal policies. In contrast, off-policy RL strategies [71] differentiate between a behavior policy for data collection and a target policy for evaluating actions' expected returns. This separation allows the behavior policy to explore various states and actions, thereby enriching the data for updating the target policy without direct impact. Consequently, off-policy methods offer a robust framework for evaluating the value of specific states and actions, enhancing their suitability for diverse environments.

Online & Offline RL: Online RL processes involve agents dynamically acquiring data through direct interaction with the environment, utilizing immediate experiences or a replay buffer to inform policy updates. Conversely, offline RL [17] precludes agent-environment interaction during training, relying instead on a predefined dataset $\mathcal{D} = \{(s, a, s', r)\}$ generated by various policies. This approach necessitates constraining the policy closely to the dataset's distribution through modern techniques [72], [73], [74] to mitigate the risk of out-of-distribution action overestimation.

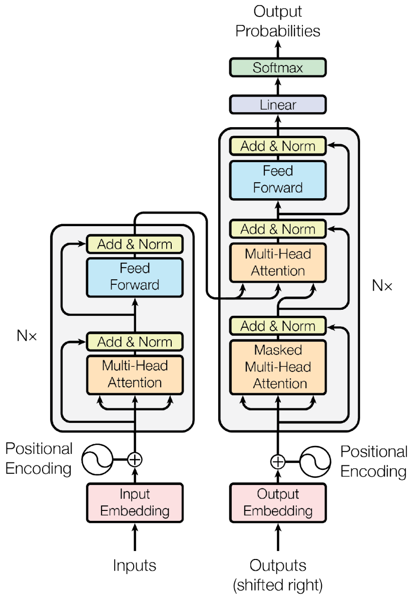

Fig. 3. Structure of the original transformer (image from [75]).

B. Transformer Architecture

The transformer [75] was first proposed as a network architecture for machine learning tasks in the field of NLP, which consists of an encoder and a decoder using stacked self-attention and point-wise fully connected layers (Fig. 3). The encoder transforms an input sequence of tokens – discrete units of text or data – into latent representations, which the decoder then sequentially processes to generate output in an auto-regressive fashion, incorporating previously generated outputs to inform subsequent ones. In the following, we describe the main components of the original transformer. For advanced transformer architectures and their variants, please refer to [76], [77], [78], along with the cited works therein.

1) Self-Attention: Self-attention [8], [79], which is the core component of the transformer, models the pairwise relations between input tokens. To calculate the self-attention, the input token representation $x \in \mathbb{R}^d$ is linearly mapped to three vectors: a query vector $q = xW_q \in \mathbb{R}^{d_q}$, a key vector $k = xW_k \in \mathbb{R}^{d_k}$, and a value vector $v = xW_v \in \mathbb{R}^{d_v}$ with dimensions $d_q = d_k = d_v$. Then, vectors derived from distinct inputs are aggregated into three matrices: Q, K, and V, facilitating the computation of the attention mechanism as delineated below (and shown in Fig. 4 left):

1) Compute the interaction scores with $S = Q \cdot K^T$;
2) Normalize S for gradient stability via $S_n = S/\sqrt{d_k}$;
3) Convert normalized scores to probabilities using the softmax function via $P = \text{softmax}(S_n)$;
4) Generate the weighted value matrix with $Z = V \cdot P$.

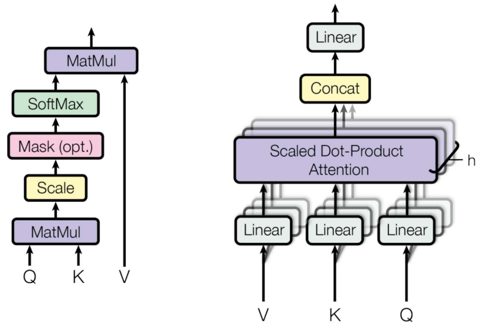

Fig. 4. (Left) Illustration of the self-attention mechanism. (Right) Depiction of multi-head attention. The image is adapted from [75].

The process can be unified into a single function:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{2}$$

The rationale underlying (2) is straightforward. Initially, it calculates interaction scores between vector pairs, which dictate the attention allocated to each word during the encoding process. Subsequently, these scores are normalized to bolster gradient stability, thereby facilitating more effective training. The normalization process transitions the scores into probabilities, ensuring that each value vector is adjusted by the cumulative probabilities. Consequently, value vectors are weighted by the aggregated probabilities, with vectors of higher probabilities receiving augmented emphasis in subsequent computational layers.

To boost the performance of the vanilla self-attention layer, the multi-head attention technique, which allows the model to jointly compute the input representation tokens from different sub-spaces, is widely used. In particular, given an input vector and the number of heads h, all inputs are first linearly mapped into three groups of vectors and packed together: the query group matrix $\{Q_i\}_{i=1}^h$, the key group matrix $\{K_i\}_{i=1}^h$, and the value group matrix $\{V_i\}_{i=1}^h$. Each group contains h vectors with dimensions $d_{q'} = d_{k'} = d_{v'} = d_{model}/h$, and the multiattention process is shown as follows:

$$\begin{aligned} &\text{MultiHead}(Q', K', V') = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^0, \\ &\qquad\quad \textbf{where } \text{head}_i = \text{Attention}(Q_i, K_i, V_i), \end{aligned} \tag{3}$$

where Q', K', V' are the concatenations of the corresponding group matrices $\{Q_i\}_{i=1}^h$, $\{K_i\}_{i=1}^h$, and $\{V_i\}_{i=1}^h$, respectively, and W^0 is the projection weight.

2) Feed-Forward Network: A feed-forward network (FFN) is a position-wise fully connected network that is applied after the self-attention layer. It consists of two linear transformation layers with a nonlinear activation function between them:

$$\text{FFN}(X) = W_2\sigma(W_1X), \tag{4}$$

where W_1 and W_2 are the learned parameters of the two linear transformation layers, and σ denotes a nonlinear activation function, such as the Gaussian error linear unit (GELU [80]) function or the rectified linear unit (ReLU [81]) function.

3) Positional Encoding: Since the input sequence is ordered and the preceding process is invariant to the position, a positional encoding with dimensionality d_{model} is added to the original input embedding. The positional embedding of the vanilla transformer is encoded as:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{\text{model}}}), \tag{5}$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}}), \tag{6}$$

where pos is the position and i is the dimension. In this way, each dimension of the positional encoding corresponds to a sinusoid, which makes it easy for the model to learn to apply attention based on relative positions and makes it possible to extrapolate to a longer sequence during the inference process.

III. Transformer-Based RL

TRL aims to harness the transformer architecture's robust representational capabilities to enhance decision-making processes in RL tasks, which has attracted much attention in recent years. Considering the success of the transformer architecture in domains where the sequential information process is critical to performance, it is an ideal candidate architecture for partially observable RL problems, where the critical observations often span the entire episode. While current RL research frequently employs LSTM architectures for agent memory, transformers offer superior feature representation for both agents and environments. However, these approaches primarily view transformers as a means for architectural improvement, remaining constrained by traditional RL algorithmic challenges, such as bootstrapping and the "deadly triad".

Due to the sequential decision process of RL, it is possible to directly involve the transformer architecture in the decision-making process and eliminate the limitations of traditional RL frameworks, which is achieved by treating the sequential decision-making process of RL problems as a sequence modeling process. Within the behavior cloning paradigm, this approach capitalizes on the transformer's long-sequence modeling capabilities, circumventing traditional RL limitations such as the necessity for regularization, conservatism, and discounting of future rewards. The utilization of simple yet effective transformer structures, rivaling the efficacy of traditional RL algorithms rely on intricate optimization processes, has attracted considerable interest from the RL community. Further investigations have been conducted to enhance performance, extend applicability to diverse tasks, and dissect the factors contributing to the successes and failures of these methods.

In this section, we delineate two distinct methodologies within TRL: Architecture Enhancement and Trajectory Optimization, as outlined in Sections III-A and III-B respectively. These methodologies are distinguished by their algorithmic characteristics and the application of transformer architectures, as depicted in Fig. 1.

A. Architecture Enhancement

In this subsection, we consider how to apply the powerful transformer structure to RL problems under the traditional RL framework (shown in Fig. 5). According to the specific part of the process in which the transformer participates, we further divide this subsection into two parts: Feature Representation and Environment Representation. For Feature Representation, we mainly introduce the methods employing transformers to derive feature representations from multi-modal inputs, enabling agents to

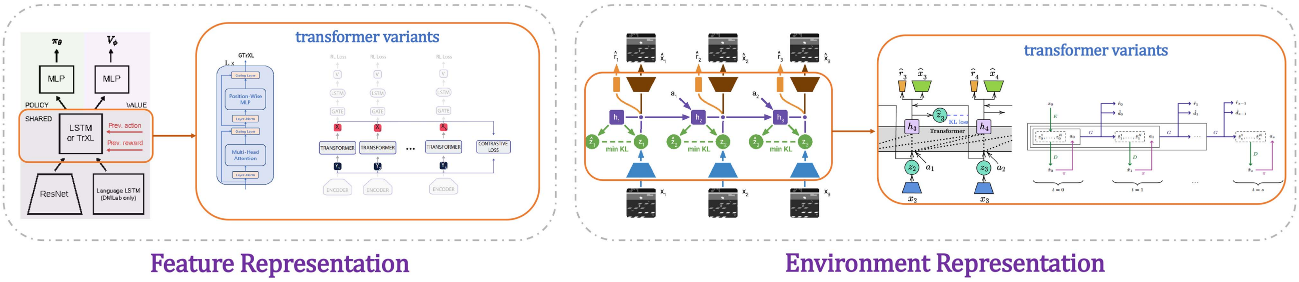

Fig. 5. A generic framework for transformer in Architecture Enhancement. *Left*: Instances where the transformer architecture supplants traditional LSTM modules to augment feature representation [19], [29]. *Right*: Implementations leveraging transformers to refine environmental representations, replacing conventional RNN-based world models [30], [32]. This bifurcation illustrates the transformer's versatility in enhancing both feature and environment modeling.

make informed decisions via value- or policy-based approaches. For Environmental Representation, the focus shifts to utilizing the transformer architecture for modeling environmental dynamics and rewards, thereby facilitating the generation of enhanced decision-making processes with additional planning algorithms.

1) Feature Representation: In the context of partially observable MDP (POMDPs), the necessity of incorporating historical observations for optimal action selection is paramount, as reliance on merely current observations proves inadequate [82]. Traditionally, gated recurrent neural networks (RNNs) like LSTM [83] and gated recurrent units (GRUs) [84] have been preferred for endowing agents with memory capabilities, superseding direct conditioning on a fixed number of recent observations [85], [86]. However, the advent of transformers, which have demonstrated superior performance over gated RNNs across various domains [87], [88], prompts an inquiry into their integration within the RL domain. Addressing this, Loynd et al. [28] proposed the working memory graph (WMG), leveraging the transformer architecture for RL. The WMG processes a variable number of factor vectors, facilitating dynamic input handling, and connects these processed features to an actor-critic network. Furthermore, it introduces ”Memos” for historical information storage, forming the backbone of its novel shortcut recurrence structure.

Concurrently, in response to the challenges in POMDP, Parisotto et al. [19] explored the potential of the transformer architecture for enhancing RL algorithms. Recognizing the instability and optimization difficulties associated with the canonical transformer model [21], they innovated upon the standard design to create the gated transformer-XL (GTrXL) [87]. This variant introduces significant improvements in stability and learning efficiency, as demonstrated in Fig. 6. Key modifications include the substitution of traditional residual connections with gating mechanisms and the strategic relocation of layer normalization to the residual pathway's ”skip” stream, a technique termed identity map reordering [89], [90]. The GTrXL offers diverse gating options, including highway connections [91], sigmoid-tanh (SigTanh) gates [92], and GRU gates [84], to optimize performance. Empirical evaluations on the DMLab-30 benchmark [93] revealed that replacing LSTM-based agents with the GTrXL network within the V-MPO algorithm [94] yielded superior results, particularly with the GRU-type gating layer outperforming across metrics.

Building on the GTrXL's success as an LSTM alternative in RL, subsequent research has focused on developing advanced memory transformers for RL applications. Rae et al. [23] expanded upon Transformer-XL [87], proposing a compressive transformer that retains rather than discards old memories, facilitating its use as an effective replacement for LSTM within the IMPALA framework [85]. This approach enhances the utility of past observations, showcasing the potential of transformers in memory compression and recall tasks. Irie et al. [22] identified the equivalence between linear transformers and the outer product-based fast weight programmers (FWPs) [95], recognizing the broader applicability of the original FWP formulation. By incorporating recurrent connections into FWP, they developed the recurrent FWP (RFWP), achieving significant performance improvements in the standard Atari 2600 domain [96] with both compact and scalable models. Mao et al. [97] further introduced transformer-based networks tailored for RL, which is agnostic to the training RL algorithm and provides off-the-shelf backbones for most RL settings.

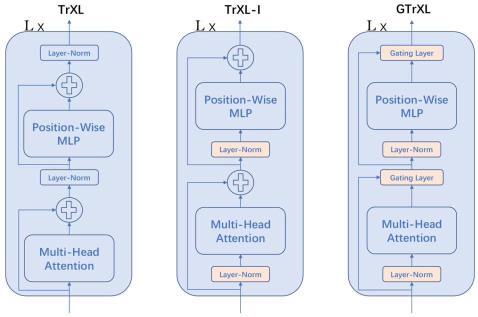

Fig. 6. Transformer variants. **Left:** Standard Transformer(-XL) architecture. **Center:** TrXL-I modifies the architecture by applying layer normalization to the input stream of the sub-modules. **Right:** GTrXL innovates by substituting the residual connection with a gating layer.

While LSTM excels at capturing recent dependencies, its capacity for processing long-range dependencies is outmatched by transformers. Recognizing this, Banino et al. [29] introduced CoBERL, an innovative agent that synergizes LSTMs with transformer architecture. This configuration leverages the transformer's proficiency in managing long contextual dependencies to augment the LSTM, thereby optimizing memory utilization [98]. CoBERL's effectiveness was validated through experiments within both on-policy and off-policy frameworks using the DeepMind Control Suite [99] and DMLab-30 [93], employing V-MPO [94] and R2D2 [86] algorithms, respectively. Above all, these advancements underscore the transformative impact of transformer architectures in RL, addressing memory management, computational efficiency, and generalization challenges, thereby motivating continued exploration of transformer-based solutions for feature representation.

2) Environmental Representation: Model-based RL methods have garnered interest for their high data efficiency. Recent innovations have explored the application of world models in RL, spanning pure representation learning [100], look-ahead search [101], and imaginative learning frameworks [102], [103]. Among these, the Dreamer agent [103], a proponent of learning within an imaginative construct, has notably advanced the field. Building on Dreamer's foundation, Chen et al. [30] introduced TransDreamer, which integrates a more potent transformer architecture into the RNN-centric Dreamer approach, marking a significant leap with the creation of the first transformer-based stochastic world model, the Transformer State-Space Model (TSSM). This model supersedes the RNN in the RSSM [102], offering enhanced stochastic action-conditioned transitions while retaining the transformer's inherent parallel computation capabilities. Empirical comparisons reveal TransDreamer's superiority over Dreamer in scenarios demanding intricate, long-term memory interactions, underscoring the TSSM's advanced predictive capabilities.

Furthering the exploration of transformers in imagination-based learning, Micheli et al. [32] proposed Imagination with auto-regression over an inner speech (IRIS), an agent that operates within an imaginative realm constructed by a discrete auto-encoder and a GPT-like auto-regressive transformer. This innovative method translates dynamic learning into a sequence generation task, with the transformer simulating environment dynamics. IRIS, trained exclusively on imagined interactions, demonstrated remarkable efficiency over other sample-efficient RL methods on the Atari 100 k benchmark [104], employing the DreamerV2 [103] framework for the policy learning process. Moreover, various methodologies have emerged, employing auto-encoders to encode environment actions and states into discrete latent variables, with transformer-based transition models trained on these variables. Ozair et al. [105] used a Monte Carlo tree search (MCTS [106]) to plan future actions and observations, while Sun et al. [31] adopted a beam search [107] to reduce performance degradation.

In model-based RL, learning a reward function is as pivotal as mastering transition dynamics. Task-specific reward functions, crucial for steering agents towards specialization, are traditionally crafted by humans, a task that proves challenging in complex or real-world scenarios. Addressing this challenge, Fan et al. [33] pioneered MINECLIP, utilizing transformer technology to autonomously generate reward signals from video demonstrations. The MINECLIP model, through $\Phi_R : (G, V) \rightarrow \mathbb{R}$, evaluates the congruence between linguistic goals and visual observations, optimizing behavior alignment via the InfoNCE objective [110]. MINECLIP's ability to produce linguistically aligned, high-reward outputs demonstrates its effectiveness as an autonomous metric for complex tasks, showcasing the potential of transformers to model adaptable reward function in RL. To sum up, these advancements reveals transformers' capacity to significantly enhance model-based RL methodologies, both in transition model sophistication and reward function adaptability, setting a new and promising direction for future research in the field.

B. Trajectory Optimization

The conceptualization of RL as a conditional sequence modeling problem was initially advanced by the DT [24] and TT [25] algorithms. These foundational efforts have spurred a series of subsequent research (illustrated in Fig. 7). This subsection outlines the DT and TT algorithms, discussing their inherent limitations and conditions for optimality. It then explores the integration of sequence modeling techniques with traditional RL principles, highlighting the resultant performance enhancements. The discussion extends to the adaptation of pretraining mechanisms, common in NLP and CV, to RL, underscoring the advantages of this incorporation within the training framework. Further, we examine algorithms enabling a single agent to excel across various, including unseen, environments. The application of transformer-based approaches to multi-agent RL systems is also presented, demonstrating the scalability of transformers from single-agent scenarios. We summarize the results of a subset of these methods in Tables II and III to demonstrate the development of transformer-based trajectory optimization methods. Despite occasional convergence to sub-optimal solutions, the employment of transformer architecture and conditional sequence modeling significantly improve RL agents' scalability and generalization capabilities, marking a significant advancement in the field's evolution.

1) Conditioned BC: Chen et al. [24] innovatively applied the DT method to offline RL, conceptualizing policy learning as a sequence modeling problem (shown in Fig. 8(a)). This method models trajectories as sequences of state, action, and returns-to-go (RTG) tuples over time steps, suitable for auto-regressive training and generation:

$$\tau = (\widehat{R}_1, s_1, a_1, \widehat{R}_2, s_2, a_2, \ldots, \widehat{R}_T, s_T, a_T), \tag{7}$$

where RTG $\widehat{R}_t = \sum_{t'=t}^{T} r_{t'}$, denoting cumulative rewards from a current time step to an episode's end, directs generated actions towards future returns, offering a more forward-looking approach than focusing on immediate rewards. Training involves trajectory sampling from offline datasets, optimizing against cross-entropy loss for discrete actions or mean squared error for continuous ones. During evaluation, this approach initiates the model with a specific target return based on the desired performance on a given task as well as the environmental state. Then, the DT selects an action, observes the new state and reward, updates the RTG value by subtracting the observed reward, and repeats this procedure until episode termination. Distinct from mere behavior cloning, the DT method excels in sparse reward environments and mitigates stability issues associated with bootstrapping and long-term credit assignment [111], [112]. This advancement not only demonstrates the efficacy of transformers in RL but also underscores their potential to address longstanding challenges in policy learning, thereby setting a precedent for future transformer-based RL research.

Concurrently, Janner et al. [25] introduced the TT, employing the transformer architecture to delineate trajectory distributions, while integrating model-based elements by modeling state and reward transitions and discretizing each dimension independently. The TT framework, visualized in Fig. 8(b), represents trajectories by discretizing N-dimensional states and M-dimensional actions as follows:

$$\tau = (s_t^1, s_t^2, \ldots, s_t^N, a_t^1, a_t^2, \ldots, a_t^M, r_t)_{t=1}^{T}. \tag{8}$$

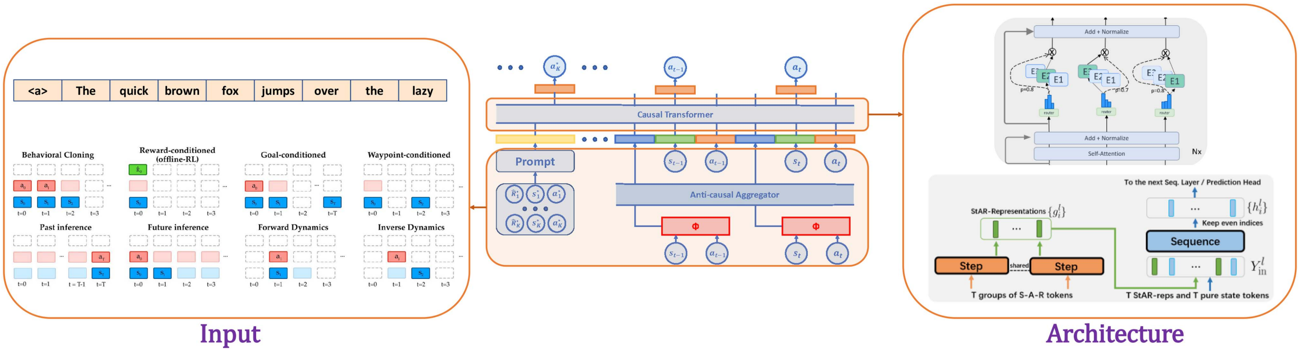

Fig. 7. A generic framework for transformer in Trajectory Optimization. The middle structure is a combination of Prompt-DT [46] and GDT [45], encompassing both input and architectural components. Diverse objectives are attained by alterations to the input format (e.g., left part, pretraining on language datasets [40] or utilizing randomly masked pretraining [108]) and transformer architecture modifications (e.g., right part, enhancing performance with a sparsely activated switch layer [109] or integrating an inductive bias layer [37]).

TABLE II
COMPARISON OF DIFFERENT TRANSFORMER-BASED TRAJECTORY OPTIMIZATION METHODS ON THE D4RL GYM ENVIRONMENTS

Gym Tasks	BC [24]	StAR [37]	QDT [38]	DT [24]	ChibiT [40]	V-ADT [113]	MTM [43]	TT [25]	Graph-DT [114]	MO-TRDT [115]	BooT [116]	DC [117]	CGDT [39]
halfcheetah-medium-expert-v2	59.9	93.7	79	86.8	91.7	91.7	94.7	95	93.2	94.4	94	93	93.6
hopper-medium-expert-v2	79.6	111.1	94.2	107.6	110	101.6	112.4	110	111.1	111.5	102.3	110.4	107.6
walker2d-medium-expert-v2	36.6	109	101.7	108.1	108.4	112.1	110.2	101.9	107.7	108.5	110.4	109.6	109.3
halfcheetah-medium-v2	43.1	42.9	42.3	42.6	43.3	48.7	43.6	46.9	42.9	43.2	50.6	43	43
hopper-medium-v2	63.9	59.5	66.5	67.6	82.1	60.6	64.1	61.1	77.1	65.9	70.2	92.5	96.9
walker2d-medium-v2	77.3	73.8	67.1	74	77.8	80.9	70.4	79	76.5	75.9	82.9	79.2	79.1
halfcheetah-medium-replay-v2	4.3	36.8	35.6	36.6	39.7	42.8	43	41.9	40.5	42.6	46.5	41.3	40.4
hopper-medium-replay-v2	27.6	29.2	52.1	82.7	81.3	83.5	92.9	91.5	85.3	97.8	92.9	94.2	93.4
walker2d-medium-replay-v2	36.9	39.8	58.2	66.6	71.3	86.3	77.3	82.6	77.5	86.9	87.6	76.6	78.1
Average	47.7	66.2	66.3	74.7	78.4	78.7	78.7	78.9	79.1	80.7	81.9	82.2	82.4

TABLE III
A SUMMARY OF TRANSFORMERS FOR TRAJECTORY OPTIMIZATION METHODS

Method	Setting	Sequence	Prediction	Hindsight Info	Inference	Additional Structure/Usage
DT [24]	Model-free; Offline	rtg-s-a	a	rtg	conditioning	basic Transformer structure
TT [25]	Model-based; IL/GCRL/Offline	s-a-r(-rtg)	s-a-r	rtg	beam search	basic Transformer structure
DC [117]	Model-free; Offline	rtg-s-a	a	rtg	conditioning	additional Convformer structure
MO-TRDT [115]	Model-free; Offline	rtg-s-a-$\bar{a}$	rtg-s-a-$\bar{a}$	rtg	conditioning	additional action region representation
ESPER [34]	Model-free; Offline (stochastic)	s-a-$\psi(\tau)$	a	expected return	conditioning	adversarial clustering
RCSL [35]	Model-free; Offline	rtg-s-a	a	rtg	conditioning	theoretical support
ODT [36]	Model-free; Online finetune	rtg-s-a	a	rtg	conditioning	trajectory-based entropy
StARformer [37]	Model-free; IL/Offline	s-a-r(-patch)	a	reward	conditioning	Step & Sequence Transformer
Graph-DT [114]	Model-free; Offline	graph rtg-s-a	a	rtg	conditioning	Graph Transformer
BooT [116]	Model-based; Offline	s-a-r-rtg	s-a-r-rtg	rtg	beam search	data augmentation
QDT [38]	Model-free; Offline	$\psi(\tau)$-s-a	a	relabelled rtg	conditioning	additional Q func
DoC [124]	Model-free; Offline (stochastic)	s-a-r-$\psi(\tau)$	a	latent feature	conditioning	additional latent value func
CGDT [38]	Model-free; Offline	rtg-s-a	a	expected returns	conditioning	additional critic network
V-ADT [113]	Model-free; Offline	$\psi(\tau)$-s-a	a	learned value	conditioning	high- & low-level transformer
ChibiT [40]	Model-free; Offline	rtg-s-a	a	rtg	conditioning	pretraining on language
MaskDP [42]	Model-free; Offline	s-a	s-a	goal/prompt	conditioning	random masking pretraining
MTM [43]	Model-free; Offline	rtg-s-a	rtg-s-a	rtg	conditioning	conditioning pretraining
MGDT [44]	Model-free; Offline	s-rtg-a-r	rtg-a-r	rtg	conditioning	multi-task learning
Gato [11]	Model-free; Offline	s-a	s-a	prompt	conditioning	generalist policy
GDT [45]	Model-free; Offline	$\psi(s,a)$-s-a	a	arbitrary	conditioning	anti-causal aggregator
Prompt-DT [46]	Model-free; Offline	rtg-s-a	a	prompt	conditioning	additional prompts from target tasks
MADT [47]	Model-free; Online finetune (multi-agent)	s-a	a	none	conditioning	separate models for actor and critic
MAT [48]	Model-free; Online (multi-agent)	s	a	none	conditioning	separate models for actor and critic
CommFormer [49]	Model-free; Online (multi-agent)	s	a	none	conditioning	additional comm. graph learning
MaskMA [50]	Model-free; Offline (multi-agent)	s	a	none	conditioning	additional mask-based training

During training, the TT maximizes the log-likelihood of each token acquired from the sequence in an autoregressive manner. Once the trajectory distribution is learned, the beam search algorithm [107] is employed together with the RTG signal to find the reward-maximizing behavior. Distinct from the Decision Transformer (DT) which directly applies predicted actions in planning, thus aligning more closely with model-free methods, the TT's comprehensive prediction of state, action, and reward tokens, coupled with beam search planning, positions it firmly within model-based methodology. This contrast underscores the nuanced roles transformers can play in RL, from enhancing direct action planning in model-free contexts to enriching model-based strategies through detailed trajectory modeling and advanced planning techniques.

While the DT approach, simplifying RL to a supervised prediction task, has gained popularity for its simplicity and efficacy across tasks, its applicability in stochastic environments is limited due to the variability of returns for given trajectories.

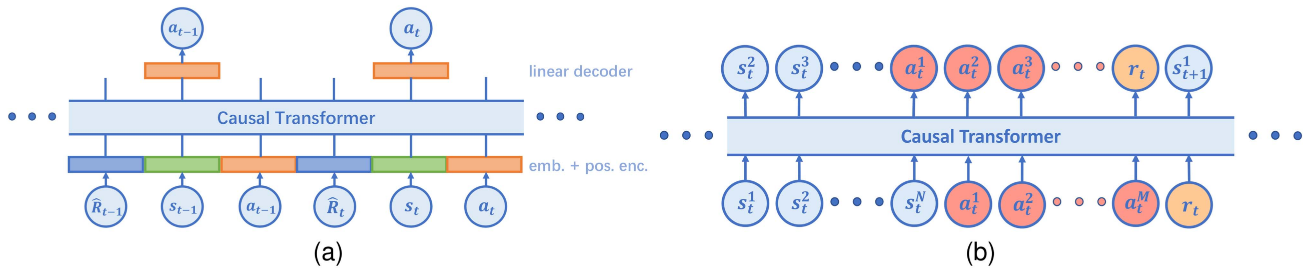

Fig. 8. (a) Decision transformer architecture: states s_t, actions a_t, and returns-to-go $\hat{R}_t$ are input into the causal transformer which outputs the next timestep actions. (b)Trajectory transformer architecture: each dimension of states s_t, actions a_t, and rewards r_t is input into the causal transformer which outputs the next dimension of states, actions, and rewards.

Paster et al. [34] highlighted that DT-like methods, relying on conditioning probabilistic models on desired returns, falter in such unpredictable settings, achieving optimal policies only when training objectives are decoupled from environmental randomness. To address this, Paster et al. introduced ESPER, a methodology conditioned on expected returns. ESPER innovates by clustering trajectories into discrete representations $I(\tau)$, subsequently predicting average returns from these clusters to guide action prediction. This structure ensures ESPER's predictions are solely influenced by the actions of the agents and independent of the stochasticity of the environment. Brandfonbrener et al. [35] further found that the necessary assumptions for the DT algorithm to obtain the near-optimal policy are stricter than those required for classic dynamic programming (DP) methods (e.g., Q-learning), such as nearly deterministic dynamics, knowledge of the target return and a conditioning value that is consistent with the distribution of the returns in the utilized dataset. These investigations reveal that although the return-conditioned supervised learning (RCSL) framework, integrated with transformer architecture, refines the traditional RL decision-making approach, inherent limitations still obstruct the creation of optimal agents in varied settings, warranting additional exploration.

2) Canonical RL: In RL, the finite nature of offline datasets typically necessitates an online environment for policy refinement through task-specific interactions and data acquisition via exploration. However, leveraging offline data to enhance online performance often proves ineffectual without strategic planning [118]. Addressing this, Zheng et al. [36] developed the online DT (ODT), an RL framework that integrates online fine-tuning with DT's pretraining. To navigate the exploration-exploitation dilemma, ODT explicitly imposes a lower bound on the policy entropy, inspired by the max-ent RL framework [119], to encourage exploration:

$$\begin{aligned}\min_{\theta}\ & J(\theta) = \mathbb{E}_{(\hat{R},s,a)\sim\tau}[-\log \pi_\theta(a|s,\hat{R})],\\ \text{subject to}\ & H_\theta^\tau[a|s,\hat{R}] \geq \beta,\end{aligned} \tag{9}$$

where β denotes a hyper-parameter and $H_\theta^\tau[a|s,\hat{R}]$ signifies policy entropy at the sequence level, diverging from the transition-level entropy utilized in SAC [120]. Empirical results reveal that ODT significantly benefits from the fine-tuning phase, establishing its efficacy and positioning it as a competitive methodology within the D4RL benchmark [121].

Addressing the integration of Markovian-like inductive bias into transformers is crucial for enhancing RL models. The inherent causal relationships among sequences of states, actions, and rewards suggest that a conventional transformer's global attention might obscure critical connections or introduce irrelevant information, particularly due to the weak links between non-adjacent tokens [25]. To mitigate these challenges, Shang et al. [37] introduced the state-action-reward transformer (StARformer), a novel architecture combining a step transformer and a sequence transformer. The step transformer focuses on local Markovian representations within state-action-reward triples, while the sequence transformer captures long-term dependencies. Unlike DT, which necessitates meticulous RTG design during inference, StARformer effectively processes stepwise rewards without compromising performance. Additionally, Graph-DT [114] explores Markovian-like inductive bias using a dependency graph and a graph transformer, achieving performance on par with StARformer but with reduced parameter count and computational demand.

Offline RL faces two primary challenges: the finite coverage of offline datasets, limiting state-action-reward transition representation [72], and the limited amount of training data, potentially impairing model performance [122]. To address these constraints and augment data coverage, Wang et al. [116] introduced the bootstrapped transformer (BooT). This algorithm enhances training by first generating data through the model itself, then leveraging this data for further model refinement, effectively mitigating overfitting by ensuring generated data aligns with the original dataset's characteristics [123]. BooT selectively trains on high-confidence generated trajectories to maintain accuracy, where confidence is quantified as:

$$c(\tau) = \frac{1}{T'(N+M+2)} \sum_{t=T-T'+1}^{T} \log P_\theta(\tau_t|\tau_{<t}), \tag{10}$$

with N and M denoting state and action dimensions, respectively (similar to TT). Despite its great efficacy demonstrated on the D4RL benchmark [121], the pseudo data generation step prolongs the total training time.

In offline RL, the capacity to synthesize optimal policies from sub-optimal trajectories – termed ”stitching ability” – is crucial [121]. This capability, inherent to traditional RL techniques like Q-learning, is notably absent in DT approaches, primarily due to discrepancies between sampled target returns and the optimal returns from actions, where high-return trajectories may not necessarily indicate superior actions but rather fortuitous outcomes [39]. To enhance DT methods' stitching ability, a spectrum of innovations has been introduced. To address the stitching ability of DT-like methods, a series of works have been proposed. Yamagata et al. [38] devised the Q-learning DT (QDT), integrating Q-learning to enrich DT by relabeling RTG tokens within the training dataset, thereby refining data quality. The DoC [124] method conditions policy on a latent future trajectory representation, reducing mutual information to effectively anticipate possible outcomes. Furthermore, approaches like the EDT [125] and CGDT [39] optimize the trajectory by dynamically filtering the optimal trajectory according to the learned value estimator. By adopting probabilistic statistics across diverse trajectories, these strategies enable policy refinement based on aggregated estimated returns, presenting a viable avenue for navigating sub-optimal data challenges.

3) Pretraining: In the domains of NLP [3] and CV [5], large-scale pretraining models have demonstrated remarkable success, capitalizing on pretraining to mitigate the computational demands posed by expressive models like transformers. The application of sequence modeling to RL invites the utilization of pretraining techniques; however, RL suffers from a scarcity of expansive datasets for pretraining purposes [126], [127]. Reid et al. [40] bridge this gap by showcasing the efficacy of leveraging natural language pretraining to enhance convergence speed and policy performance in offline RL tasks that are unrelated to language (e.g., MuJoCo [128]), providing a unified view of the sequence modeling domain. This approach employs a similarity-based objective to align language embeddings with trajectory input representations:

$$\mathcal{L}_{cos} = -\sum_{i=0}^{3N} \max_{j} \mathcal{C}(I_i, E_j), \tag{11}$$

where E_j and I_i represent language embeddings and trajectory input tokens, respectively, with $\mathcal{C}(z_1, z_2)$ denoting the cosine similarity between vectors. Similarly, Li et al. [41] also leveraged pretrained language models (LMs) to scaffold learning and generalization across sequential decision-making tasks. Through comprehensive ablation studies on BabyAI [129] and VirtualHome [130], they demonstrated the non-necessity of translating historical states and actions into natural language to reap the benefits of language pretraining. Instead, the essence of these improvements lies in the strategic encoding of sequential input and the application of language pretraining.

Beyond leveraging external language datasets, self-supervised pretraining via masking techniques on diverse unlabeled RL data has garnered significant attention. Liu et al. [42] introduced masked decision prediction (MaskDP), employing random masking of state and action tokens to enhance agent generalizability through unsupervised data. They underscored the critical role of mask ratios in pretraining for optimizing downstream task performance, attributing this to the temporal correlations within state-action sequences. Similarly, Wu et al. [43] demonstrated that training transformers with varied masking patterns equips models with versatile capabilities, adaptable for different roles by simply choosing appropriate masks at inference time. These achievements underscore the efficacy of masked pretraining in fostering adaptable and capable transformer networks for downstream RL tasks.

Beyond the expansion of pretraining datasets, identifying the most effective unsupervised objectives to enhance downstream task performance in RL is also crucial. Yang et al. [131] utilized the D4RL [121] offline dataset, focusing on three downstream task categories: limited-data imitation learning, offline RL, and online RL. They employed a transformer-based architecture pre-trained via a contrastive loss on subsets masked randomly, subsequently adapting different algorithms for each task category–BRAC [132] for offline RL and SAC [120] for online RL. Their findings underscore the positive impact of unsupervised pretraining on policy learning performance. Notably, the optimal representation learning objective for pretraining varies according to the specific downstream task, indicating the absence of a universally superior objective.

4) Generalist Agents: Traditional RL agents, often constrained by small-scale models tailored for singular tasks, face challenges in developing general agents capable of mastering a broad spectrum of tasks from extensive datasets [133]. Initial endeavors to tackle the entire Atari suite [134] via deep Q-learning [67] and actor-critic methods [135] necessitated distinct training sessions per game. Subsequent efforts aimed at training a unified neural network to concurrently engage with multiple Atari games [85], marking a shift towards multi-task learning. Lee et al. [44] extended this multi-game learning approach by evaluating the scalability and performance of various methods, including DT [24], DQN [71], and CQL [73]. Their investigation revealed that DT-based sequence modeling excelled in scalability and efficacy. However, the challenge of generating consistently expert-level actions from mixed-quality datasets persisted. To address this, Lee et al. introduced a novel inference-time methodology, which employs a binary classifier $P(\text{expert}^t|\ldots)$ and utilizes Bayes' rule to infer expert-level returns and actions:

$$P(R^t|\text{expert}^t, \ldots) \propto P_\theta(R^t|\ldots)P(\text{expert}^t|R^t, \ldots), \tag{12}$$

with the expert likelihood proportional to future returns in a manner akin to probabilistic inference [136]. Empirical validation on 46 Atari games underscored that the scaling rule can also be adapted to RL: model performance and adaptability to new tasks improve with increased model size.

Concurrently, Reed et al. [11] embarked on creating a transformer-based general agent, Gato, leveraging offline data to operate across multi-modal, multi-task, and multi-embodiment domains. Gato's design principle is to train on a diverse array of data, aiming for maximal relevance and breadth. The agent standardizes multi-modal data into a uniform sequence of tokens, upon which the transformer model predicates to predict the distribution of subsequent tokens through the following training loss:

$$\mathcal{L}(\theta) = -\sum_{b=1}^{|\mathcal{B}|}\sum_{l=1}^{L} m(b,l) \log p_\theta(s_l^{(b)}|s_1^{(b)}, \ldots, s_{l-1}^{(b)}), \tag{13}$$

with $\mathcal{B}$ representing batch sequences and $m(b,l) = 1$ for tokens derived from text or agent actions. Note that the training dataset exclusively comprises near-optimal agent experience in both simulated and real-world environments, and Gato needs the extra information prompted by expert trajectories to infer the next token during evaluation [137], [138], making it different from the work of Multi-Game DT [44].

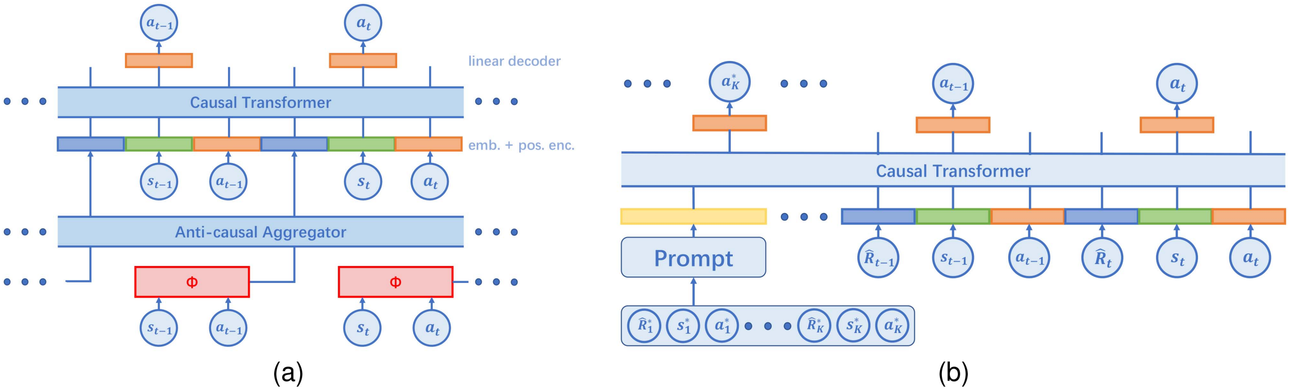

Fig. 9. (a) The GDT architecture [45] offers a versatile extension of the DT model, incorporating an anti-causal aggregator and a feature function Φ to substitute the traditional RTG token. Varied pairings of aggregators and feature functions enable the formulation of distinct algorithmic approaches. (b) The Prompt-DT architecture [46] enhances decision-making by incorporating trajectory prompt augmentation alongside historical sequences, facilitating the auto-regressive prediction of actions for given states in unseen environments.

Exploring the potential of leveraging diverse types of hindsight information for generalized decision-making in RL, Furuta et al. [45] have shown that various algorithms, such as hindsight multi-task RL [139] and upside-down RL [111], which condition on future trajectory details, can be integrated via hindsight information matching (HIM). They formalize these approaches as the information matching (IM) challenge, aiming to develop a conditional policy $\pi(a|s, z)$ that minimizes the divergence between the statistics of the generated trajectory and the desired information z:

$$\min_{\pi} \mathbb{E}_{z\sim p(z),\tau\sim\rho_z^\pi(\tau)}[D(I^\Phi(\tau), z)], \qquad (14)$$

where $I^\Phi(\tau)$ denotes the information statistics of the trajectory τ, such as identity or the reward function $r(s, a)$. To accommodate varying $I^\Phi(\tau)$, they propose the GDT, a flexible framework that adjusts its feature function and anti-causal aggregator accordingly (Fig. 9(a)). This method has demonstrated effective generalization across unseen (and even synthetic) multi-modal reward or state-feature distributions, underscoring the transformative impact of incorporating broader hindsight information with transformer into RL policy learning.

Addressing the challenge of generalizing RL to unseen tasks, traditional approaches like offline meta-RL, exemplified by MAML [140] and MACAW [141], aim for rapid adaptation via algorithmic design. Inspired by the prompt-based framework for adaptation in NLP [142], Xu et al. [46] proposed the prompt-based DT (Prompt-DT), leveraging inductive architectural bias for enhanced flexibility. Instead of utilizing a text description as a prompt, which requires much human labor to annotate [143], Xu et al. adopted trajectory segments as prompts and forced the agent to imitate these demonstrations without fine-tuning (shown in Fig. 9(b)). Evaluations in meta-RL environments (e.g., Cheetah-dir [140] or Ant-dir [141]) showed that given few-shot trajectories, the Prompt-DT is able to beat strong offline meta-RL baselines and can be generalized to out-of-distribution tasks. Hu et al. [144] further introduced a prompt-tuning technique to enhance the refinement of prompts, effectively reducing the dependence on pre-collected expert trajectories.

There are also several innovative approaches that explore the domain of generalization in RL from varied perspectives. The AnyMorph [145] framework enables policies to adapt to novel agent morphologies without the need for fine-tuning. AttentionNeuron [146] addresses the challenge of managing sudden, random input reordering during task execution. Transfer-DT [147] enhances agent robustness to environmental dynamics shifts through causal reasoning. Lastly, SwitchTT [109] introduces a sparsely activated model to mitigate the detrimental effects of indiscriminate parameter sharing. These methodologies collectively underscore the breadth of strategies for enhancing transformer-based RL generalization across varying contexts and challenges.

5) Multi-Agent Extension: Exploring the application of transformers in MARL [148] through sequence modeling, Meng et al. [47] pioneered the integration of offline pretraining with online fine-tuning within this domain. They introduced a large-scale dataset derived from MAPPO [149] implementations on the SMAC task [150], facilitating a novel approach to training RL systems on diverse datasets–a critical advancement for MARL, where online exploration may be constrained [151]. Subsequently, Meng et al. developed the multi-agent DT (MADT), framing MARL as a conditional sequence modeling challenge via an auto-regressive transformer architecture (Fig. 10(a)). This methodology encapsulates each agent's local state and action within the trajectory as:

$$\tau^i = (x_1, \ldots, x_t, \ldots, x_T) \ \text{ where } x_t = (s_t, o_t^i, a_t^i), \qquad (15)$$

with s_t representing the global state, o_t^i the local observation, and a_t^i the action for agent i at time t. The pretraining phase employs a cross-entropy loss:

$$L_{CE}(\theta) = \frac{1}{T}\sum_{t=1}^{T} P(a_t)\log P(\hat{a}_t|\tau_t, \hat{a}_{<t};\theta), \qquad (16)$$

aiming to align the model's output $\hat{a}_t$ with ground-truth actions a_t. However, when transitioning to an online environment, the pretrained model initially underperforms due to its tendency to replicate offline dataset actions. To mitigate this, the pretrained transformer is integrated into the actor and critic networks of the PPO algorithm [68], enhancing the model's online performance. Empirical results on SMAC tasks demonstrate MADT's superior

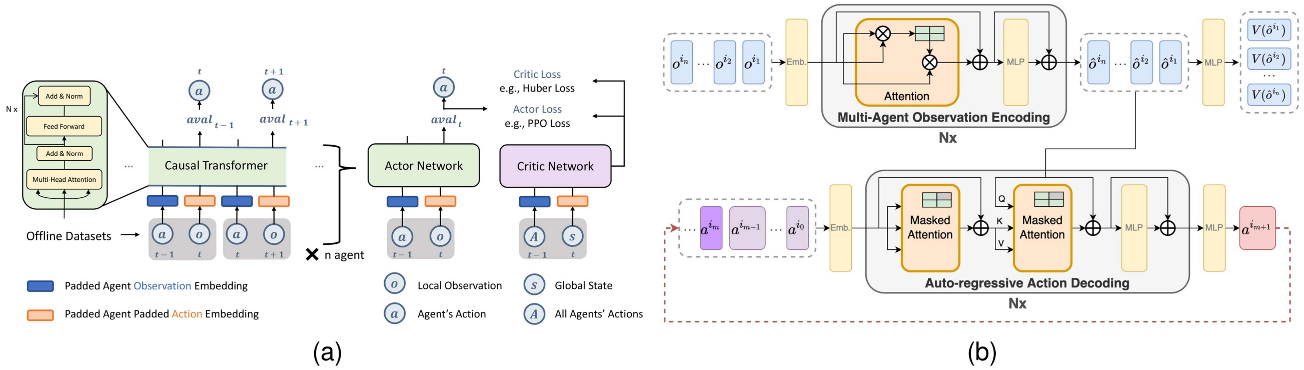

Fig. 10. (a) Architecture of MADT (image from [47]), showcasing offline pretraining (left) and online fine-tuning (right) components. (b) Encoder-decoder structure of MAT (image from [48]), with the encoder processing a sequence of agents' observations into a sequence of latent representations, followed by the decoder sequentially predicting each agent's optimal action in a sequential and auto-regressive manner.

sample efficiency and its ability to provide significant generalizability enhancements, marking a significant step forward in the application of transformers to MARL.

Concurrently, Wen et al. [48] introduced the multi-agent transformer (MAT), redefining cooperative MARL as a sequence modeling challenge through an encoder-decoder transformer structure (Fig. 10(b)). This approach addresses the complexity of agent interactions in multi-agent systems, where independent transformer policies for each agent do not necessarily lead to joint performance enhancement [152]. Inspired by the multi-agent advantage decomposition theorem [153], MAT transitions joint policy optimization into a sequential policy search, which is formulated as:

$$A_{\pi}^{i_{1:n}}(o, a^{i_{1:n}}) = \sum_{m=1}^{n} A_{\pi}^{i_m}(o, a^{i_{1:m-1}}, a^{i_m}), \tag{17}$$

offering a framework for monotonically improving performance through sequence modeling. Diverging from MADT, which relies on offline imitation learning, MAT is trained only via online trial and error through a PPO-likely objective. Extensive experiments conducted on SMAC, Multi-Agent MuJoCo [154], and Google Research Football [155] affirm MAT's superior performance, data efficiency, and effectiveness as a few-shot learner. However, MAT requires intensive agent-to-agent communication, posing challenges in resource-constrained environments. CommFormer [49] addresses these limitations by optimizing the communication graph alongside the architectural parameters, introducing a bi-level optimization strategy. By employing a continuous relaxation for graph representation and integrating attention mechanisms, CommFormer successfully facilitates an end-to-end learning process that maintains high performance with much less communication requirement.

Exploring diverse approaches within MARL through transformers, several models offer innovative solutions to domain-specific challenges. The ATM [156] model addresses partial observability by integrating memory through a transformer-based network and applying semantic inductive biases via an entity-bound action layer. UPDeT [157] designs versatile multi-agent policies by decoupling policy distribution from observations through transformers and self-attention-derived importance weights, accommodating various observation and action configurations. MaskMA [50] enhances zero-shot capabilities and navigates centralized training versus decentralized execution dilemmas by masking units and learning collaborative policies, alongside generalizing action representations to adapt to varying agent numbers and action spaces. These approaches collectively advance transformer-based MARL, tackling unique challenges across different scenarios.

IV. Applications of TRL

RL has evolved from simplistic virtual scenarios to addressing complex real-world challenges, facilitated by advancements in deep neural network architectures. This progression has expanded RL's applicability to diverse domains such as Robotic Manipulation, TBGs, Vision-Language Navigation, and Autonomous Driving. These areas, characterized by the necessity to process multi-modal inputs within intricate settings, are ideally suited for leveraging the transformer architecture's full potential. The following discussion outlines the problem formulations within these domains and explores transformer-based solutions.

A. *Robotic Manipulation*

In robotics, the development of autonomous agents focuses on automating intricate real-world tasks, necessitating the learning of RL policies for effective robotic manipulation. This process entails predicting future agent positions within an environment, incorporating contextual data for informed planning. Robotic manipulation aims to achieve human-like dexterity and decision-making, presenting ongoing challenges in spatial and temporal modeling of agent interactions.

In robotic domains, equipping robots with the human-like ability for one-shot imitation learning involves inferring intentions from visual demonstrations and applying these insights to achieve similar goals. This process requires selecting appropriate goal representations and integrating them with policy networks alongside current observations. Leveraging attention mechanisms, akin to human cognitive processes [158], has shown to enhance task performance when guiding policies with human attention [159]. Dasari et al. [51] then proposed the T-OSIL method and employed transformers to capture relational features from demonstrations and observations. To address limitations in feature weighting at test time, they introduced an unsupervised inverse dynamics loss, compelling the

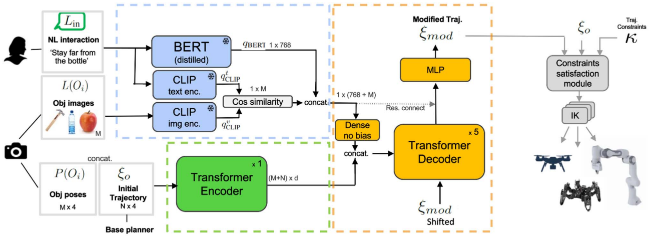

Fig. 11. The framework of LATTE (image from [52]), which includes a language and contextual encoder (blue), a geometry encoder (green), and a multi-modal transformer decoder (orange).

transformer to model dynamic interactions, thereby enhancing policy effectiveness in multi-agent MuJoCo environments [128]. However, Zhao et al. [160] critiqued previous one-shot imitation learning approaches for assuming high similarity between training and testing phases. They proposed MOSAIC, utilizing a transformer-based policy network with a temporal contrastive loss to distinguish adjacent from non-adjacent frame representations. This method seeks to refine temporal representation coherence, underscoring transformers' potential in advancing robotic imitation learning by fostering nuanced understanding and adaptation in dynamic environments.

In human-robot interaction, leveraging natural language to convey human intent to robots emerges as a highly intuitive approach. Bucker et al. [161] developed a language-based interface enabling users to adapt robotic trajectories through verbal commands. This innovation frames trajectory generation as a sequence modeling task, effectively utilizing transformer language models for semantic instruction encoding. Subsequently, a multi-modal attention mechanism aligns these instructions with geometrical trajectory data. Their findings indicate that leveraging pretrained language models with multi-modal transformers surpasses traditional methods such as kinesthetic teaching and programming-based obstacle avoidance. Further advancements by Bucker et al. [52] incorporated a CLIP image encoder [162] to extend trajectory modeling into 3D space (LATTE, Fig. 11), enhancing the system's application across various robotic forms, including aerial and legged robots. Additional research [163], [164] explores the development of versatile instruction-following agents via transformer-based RL methods, underscoring the widespread utility and transformative impact of transformers in facilitating sophisticated human-robot interactions.

In task-specific robotic settings, understanding task constraints and dynamic environments is crucial. Jain et al. [53] introduced a prompt-situation architecture, TTP, designed to adapt to user preferences and generalize to unseen preferences using a single demonstration. TTP leverages demonstration trajectories and current states to imitate embedded preferences:

$$\min_{m\sim M,\tau\sim D_m,(S,a)\sim D_m} \mathcal{L}_{CE}(a,\pi(S,\psi(\tau))) \tag{18}$$

where m represents preferences, D a multi-preference dataset, and τ the prompt containing state-action pairs. TTP's efficacy is demonstrated in simulated kitchen environments and with real-world applications using the Franka Panda robotic arm. A common use of transformers in the robotic manipulation domain is to serve as feature extraction modules for one or more modalities simultaneously.

Transformers are also increasingly utilized in robotic manipulation for feature extraction across multiple modalities. Ongoing research aims to refine transformer architectures for enhanced representation learning from various inputs, including visual, linguistic, and textual data, for tasks involving grippers [165], [166], legged locomotion [167], and dual-arm robots [168], predominantly trained via BC patterns.

B. Text-Based Games

TBGs are interactive simulations where agents engage through text-based observations and commands. Conceptually, TBGs are modeled as partially observable MDP, with agents initially unaware of the ultimate goal, navigating through the game by achieving a series of subgoals for sparse rewards. The challenge in TBGs arises from the environment's complexity, the game length, and the verbosity [169]. Frameworks like TextWorld [169], LIGHT [170], Jericho [171], and TextWorld with QA [172] have significantly contributed to TBGs' development, facilitating research and application in this domain.

Standard RL algorithms struggle with exploring expansive action spaces, a challenge exemplified in TBGs where games may present over a billion possible actions at each step [171]. Ammanabrolu et al. [173] proposed leveraging an oracle agent's expertise, introducing question-answering (QA) in TBGs to guide action selection through a network trained on oracle agent experiences. They utilized knowledge graphs as state representations, leading to the development of KG-A2C [174], marking a significant advancement in navigating combinatorial action spaces. Building on this, Ammanabrolu et al. [54] introduced Q*BERT, enhancing sample efficiency by employing the pretrained transformer model for QA tasks within TBGs. For each observation, Q*BERT generates a set of questions processed by the transformer model, with the answers updating the knowledge graph to inform subsequent actions. This approach enables Q*BERT to achieve faster asymptotic performance, evidencing the QA system's role in accelerating learning and boosting model efficiency.

Adhikari et al. [55] identified a reliance on heuristics within previous works that leverage the inherent structures of games, such as KG-A2C [174], which utilizes hand-crafted methods for updating its knowledge graph. Addressing this, they introduced the graph-aided transformer agent (GATA), innovating with a data-driven approach to construct and update graph-structured beliefs. GATA merges the transformer's capabilities with a dynamic belief graph, enhancing action selection by capturing game dynamics more effectively. GATA comprises two key components: a graph updater, which treats observation reconstruction as a sequence-to-sequence task, and an action selector that encodes the belief graph and observations into representations, refined through bidirectional attention. Demonstrating superior performance over baselines, GATA validates the advantage of transformer-based, graph-structured representations in RL. Tuli et al. [175] further augment GATA with linear temporal logic (LTL), enabling the monitoring of progress toward instruction fulfillment. This addition directs agent actions towards achieving defined goals, showcasing the transformer's role in enhancing RL agents' efficiency and goal-directed behavior with the LTL module.

Shifting focus to model-based planning, Liu et al. [56] tackled the challenge of learning dynamics models within TBGs, characterized by partially observable states and complex text dynamics. They introduced the object-oriented text dynamics (OOTD) model, employing graph representations for objects and distinct transition layers for predicting belief states, without knowing the rewards and observations during planning. Central to OOTD are transformer-based models for both state transition predictions and reward correlations, trained via object- and self-supervised methods. Demonstrated on the TextWorld benchmark [169], OOTD outperforms traditional model-free methods by significantly improving sample efficiency and overall efficacy, underscoring the potential of transformers in enhancing model-based RL strategies.

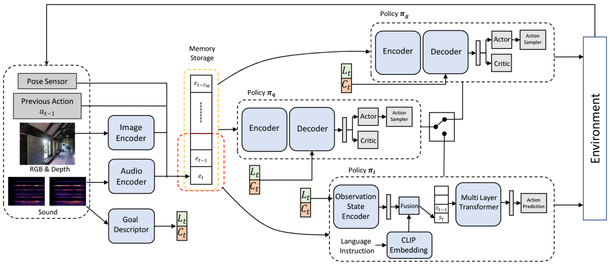

Fig. 12. Overview of the AVLEN framework [189]: This diagram illustrates the utilization of transformer architecture for encoding multiple input modalities, including audio, visual, and linguistic data. AVLEN integrates these inputs through a hierarchical RL policy structure, consisting of two distinct levels designed to process the multi-modal information efficiently.

C. Navigation

The domain of visual-linguistic navigation (VLN) has gained notable interest [176], challenging agents to interpret language instructions, perceive their surrounding environments, and execute actions to achieve specified goals. VLN tasks are modeled as partially observable MDP, wherein agents must navigate with incomplete information, relying on memory to integrate partial instructions for decision-making. To advance VLN research, various datasets and simulators have been introduced, including R2R [176], R2RIE [177], and Touchdown [178] for natural language-guided navigation, dialogue-driven methods like CVDN [179], VNLA [180], and HANNA [181], along with REVERIE [182] for object localization tasks.

The challenge of partial instruction coverage and the complex relationship between visual states and language instructions in VLN presents significant hurdles for methods learning instruction understanding from scratch [183], [184]. To address these issues, Hao et al. [57] introduced a pretraining strategy that aligns language instruction representations with visual states, mitigating instruction ambiguity. This approach comprises two primary tasks: image-attended masked language modeling (MLM), akin to BERT's pretraining, which predicts masked words within instructions using surrounding context and visual states; and action prediction (AP), which forecasts actions directly from instructions and visual inputs without trajectory history. The combined pretraining objective optimizes both MLM and AP tasks, enhancing the model's ability to interpret and act upon instructions. Further contributions, such as PRESS [185] leveraging BERT for instruction comprehension, VLN-BERT [186] fine-tuning ViLBERT [187] on instruction-trajectory pairs, and BEVBert [188] adopting pretraining with a unified map, underscore the value of pretraining in VLN. Hong et al. [58] further innovated by integrating a recurrent function into the pretrained BERT model, enhancing performance with reduced computational demands.

To handle more complex environments and surpass the limitations of recurrent states in VLN tasks, Chen et al. [59] developed the history-aware multi-modal transformer (HAMT) to encode historical information more effectively. Utilizing a cross-modal transformer, HAMT identifies long-range dependencies between current observations and instructions against historical data, integrating a hierarchical transformer structure to enhance computational efficiency. Pretraining on tasks like AP and MLM, followed by fine-tuning with an RL and imitation learning objective, HAMT demonstrates robust performance across varied environments. Chen et al. [190] further refine the navigational capabilities by developing a topological map to explicitly track all visited and navigable locations observed thus far, enabling efficient long-term navigation planning. Pashevich et al. [191] also introduced a similar multimodal transformer model (E.T.) that encodes complete episode histories, using synthetic instructions to simplify learning and enhance generalization. In a more complex endeavor, Paul et al. [189] explored the audio-vision-language environment with the transformer architecture (AVLEN, Fig. 12), aiming to localize audio sources within realistic visual settings, a step closer to real-world navigation challenges. These advancements underscore transformers' pivotal role in advancing VLN by processing complex multimodal data and adapting to dynamic environments, highlighting their significant potential for broader RL applications.

D. Autonomous Driving

The autonomous driving problem can be defined as point-to-point navigation in an urban setting while maintaining a safe distance from other dynamic agents and following traffic rules. Approaches to self-driving vehicles (SDVs) generally fall into supervised learning and RL categories. Innovations like the CARLA [192] and NoCrash [193] simulators, alongside datasets such as Nuscenes [194], have spurred developments like the SAM [195] driving method, and models like ST-P3 [196] and UniAD [197] that utilize intermediate representations for enhanced navigation. While supervised methods depend heavily on extensive labeled datasets and suffer from limited interpretability and performance constraints due to reliance on pre-defined trajectories, deep RL emerges as a promising alternative, indicating a shift towards more adaptive and interpretable autonomous driving solutions.

SDVs process information from diverse sources, notably images from cameras and point clouds from LiDAR sensors, necessitating effective sensor fusion techniques. Prakash et al. [60] introduced the TransFuser model for multi-modal fusion, utilizing a self-attention transformer to integrate different modalities and predict future waypoints auto-regressively via a GRU model. Trained through imitation learning, TransFuser aims to replicate expert trajectories, minimizing the expected loss between predicted and actual waypoints. Despite its innovations, Shao et al. [61] identified limitations in TransFuser's sensor scalability

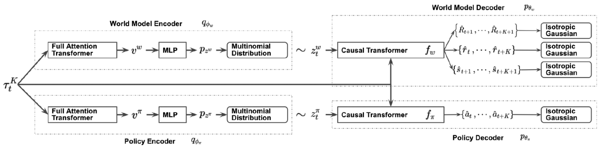
World Model Encoder q_{ϕ_w}
Full Attention Transformer
v^w
MLP
p_{z^w}
Multinomial Distribution
$\sim z_t^w$
World Model Decoder p_{θ_w}
Causal Transformer f_w
$\{\hat{R}_{t+1}, \cdots, \hat{R}_{t+K+1}\}$
$\{\hat{r}_t, \cdots, \hat{r}_{t+K}\}$
$\{\hat{s}_{t+1}, \cdots, \hat{s}_{t+K+1}\}$
Isotropic Gaussian
Isotropic Gaussian
Isotropic Gaussian
τ_t^K
Full Attention Transformer
v^π
MLP
p_{z^π}
Multinomial Distribution
$\sim z_t^\pi$
Causal Transformer f_π
$\{\hat{a}_t, \cdots, \hat{a}_{t+K}\}$
Isotropic Gaussian
Policy Encoder q_{ϕ_π}
Policy Decoder p_{θ_π}

Fig. 13. Overview of the SPLT Framework [62]: This framework features a transformer-based dual-model approach with a world model (top) tasked with reconstructing discounted returns, rewards, and states, alongside a policy model (bottom) dedicated to generating the action sequence.

and its restriction to LiDAR and single-view images, potentially hampering comprehensive environmental perception. Addressing these concerns, they developed InterFuser, an interpretable sensor fusion transformer capable of amalgamating data from multi-modal multi-view sensors for enriched feature representations. Demonstrated on the CARLA simulator, InterFuser outperforms in navigating complex and adversarial urban conditions, showcasing the transformative impact of transformers in enhancing multimodal sensor fusion and decision-making in autonomous driving.

To address the challenge of applying sequence modeling in RL to stochastic, safety-critical domains like autonomous driving, Villaflor et al. [62] introduced the separated latent trajectory transformer (SPLT, Fig. 13), innovating by disentangling policy effects from world dynamics to mitigate such risks. This separation allows for the creation of diverse behavioral predictions and a broad anticipation of environmental responses, enhancing adaptability and safety in autonomous navigation, as evidenced by superior performance on CARLA compared to traditional models [24], [74]. Addressing another prevalent issue, the assumption of unimodal expert data, Shafiullah et al. [198] developed the behavior transformer (BeT) to learn from inherently multi-modal, sub-optimal data. BeT segments continuous actions into ”action centers” and ”residual actions,” utilizing transformers for mapping observations to discrete action distributions. This approach ensures comprehensive coverage of data modes demonstrated on CARLA.

V. Discussion

Transformers are becoming hot topics in the field of RL due to their strong performance and tremendous potential, as summarized in this survey. While this discussion encapsulates key contributions, the application of transformers in RL extends beyond the scope here. Given their success across the AI spectrum, the integration of transformers with RL represents a burgeoning area of interest. Nevertheless, the potential of transformers for performing RL has not yet been fully explored, meaning that several limitations and challenges still need to be resolved and several future prospects need to be considered. This section delves into these challenges and potential advancements, providing a nuanced perspective on the integration of transformers in RL.

A. Limitations and Challenges

Local Context Sensitivity: Transformers, while adept at global context assimilation for high-level planning, struggle with capturing local contexts effectively due to their self-attention mechanism’s focus on point-wise comparisons across entire sequences [199]. Innovations inspired by CNN, such as local window-based attention [37], [114], [200], aim to enhance the balance between local detail and global awareness in decision-making processes.

RL-Specific Transformer Architectures: Current transformer applications in RL predominantly adapt NLP-focused models [75], underscoring the absence of transformers tailored for RL. For example, Parisotto et al. [19] modified the standard transformer architecture to obtain a more stable and faster training process, Shang et al. [37] tried to incorporate a Markovian-like inductive bias into a transformer, and Villaflor et al. [62] disentangled the policy and world dynamics generation processes to produce safer outcomes. These improvements are attained in a task-driven manner yet a universally applicable RL-centric transformer; thus, the creation of an RL transformer is still an open problem waiting to be solved.

Stochastic Environment Adaptation: Although sequence modeling methods such as the DT [24] have achieved comparable performance on various tasks, these works are only the first steps in the RL domain and still have much room for improvement. As shown in [34], [35], these methods have high failure probabilities in stochastic environments, and many preconditions need to be satisfied to reach the optimal policy, such as deterministic dynamics and a known target return, where the target return corresponds to the distribution of the given dataset. Although Paster et al. [34] tried to condition the expected return to mitigate the negative effect of the uncertainty in the training dataset derived from a stochastic environment, their results were still inferior to those of a traditional RL algorithm, and it is less appropriate to condition the model on the mean return when the input to the agent is a visual goal. There is still a long way to go to overcome these preconditions and limitations by changing the training method or improving the transformer structure.

Efficiency in Model Design: The computational demands of transformers, contrasted with the lightweight nature of traditional RL networks, pose a barrier to their widespread adoption in RL, especially for tasks requiring extensive contextual data [11], [44]. Although some methods try to reduce the size of their network structures through compression or distillation [23], [201], they are still computationally expensive. This challenge, compounded by hardware limitations and the constraints on model scalability, curtails the capacity of transformers to manage extended input sequences. Therefore, the development of computationally efficient transformer models is imperative to align TRL methods with the speed of conventional RL algorithms and facilitate their application in environments constrained by computational resources.

B. Future Prospects

Besides the limitations and challenges identified, this survey also proposes several avenues for future research to harness the full potential of transformers in RL.

Addressing the integration of transformers within traditional RL algorithms necessitates the resolution of several pivotal challenges. First, the substantial computational demands posed by transformers require optimization to enhance their feasibility for broader RL applications. Second, the potential of transformers to serve as adaptive memory modules, offering an alternative to LSTM units particularly under POMDP conditions, warrants further exploration. This exploration should not only focus on their capacity as memory aids but also investigate additional roles transformers could assume within the RL paradigm. Furthermore, the literature [202] suggests that transformers have the capability to emulate all RL algorithms given a sufficiently expansive and diverse training dataset. This insight opens the possibility of developing a versatile model capable of dynamically selecting and implementing the appropriate traditional RL algorithm based on specific task requirements, a concept yet to be thoroughly investigated.

Reconceptualizing RL as a sequence modeling challenge through transformers simplifies certain traditional complexities but may also diminish inherent RL advantages. A critical area of research is integrating these advantages into transformer-based sequence modeling. The DT-like approach, reliant on RTG tokens for future sequence forecasting, encounters limitations in optimizing outcomes across various environments due to inadequately set RTG conditions. Identifying optimal conditions and functional approaches for setting RTGs presents a promising research direction. Exploring the development of a versatile policy framework that processes various modalities (such as images, videos, texts, and speech) through standardized processing blocks is another area warranting attention. Despite progress in training multi-domain capable agents, their ability for efficient generalization from extensive data sets remains uncertain. Investigating methods for training agents to generalize to unseen tasks without heavy assumptions emerges as a vital research direction [147]. Furthermore, assessing transformers' capacity to develop a generalized world model [203] across tasks and scenarios is of considerable interest, signaling a pivotal shift towards more generalist AI systems.

In practical applications, in addition to using the transformer structure as a multi-modal multi-task feature extraction method, it is also worth studying how to apply the transformer structure to decision networks. Understanding the sources of errors and potential adverse impacts arising from transformer implementations is crucial for ensuring the security and reliability of real-world applications. This necessitates a comprehensive analysis of the transformer's operational intricacies and their implications for decision-making processes.

While the enhancement of RL through the adoption of transformers has been extensively discussed, the converse – utilizing RL to augment transformer training – represents a less explored yet compelling avenue. Certain studies have already embarked on this path, employing offline RL settings for language and dialogue generation tasks, adopting strategies such as relabeling [204] and value functions [205] to refine generation policies. Furthermore, the approach of reinforcement learning from human feedback (RLHF) [206] demonstrates the use of RL to adjust Transformers, specifically for aligning language models with human intent. This emerging paradigm suggests that RL could serve as a pivotal mechanism for enhancing transformer performance across various domains in future research endeavors.

VI. Conclusion

This survey offers a thorough overview of the integration of transformers in RL and their application in various real-world scenarios. Initially, we delineate RL fundamentals, focusing on its categorizations, and the transformer architecture. Subsequent sections delve into the deployment of transformers for Architecture Enhancement and Trajectory Optimization in RL, accompanied by an exploration of their application across domains such as Robotic Manipulation, Text-Based Games, Navigation, and Autonomous Driving. Finally, we discuss unresolved issues within this research area, highlighting existing challenges and limitations, and propose avenues for future investigation that promise to advance the field further.

References

[1] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, "Natural language processing: An introduction," *J. Amer. Med. Inform. Assoc.*, vol. 18, no. 5, pp. 544–551, 2011.

[2] T. Brown et al., "Language models are few-shot learners," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 1877–1901.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[4] M. Chen et al., "Generative pretraining from pixels," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1691–1703.

[5] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[6] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 9992–10002.

[7] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.

[8] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," 2016, *arXiv:1606.01933*.

[9] A. Ramesh et al., "Zero-shot text-to-image generation," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8821–8831.

[10] J.-B. Alayrac et al., "Flamingo: A visual language model for few-shot learning," 2022, *arXiv:2204.14198*.

[11] S. Reed et al., "A generalist agent," 2022, *arXiv:2205.06175*.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[13] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: A comprehensive survey," *Artif. Intell. Rev.*, vol. 55, pp. 945–990, 2022.

[14] A. Singla, A. N. Rafferty, G. Radanovic, and N. T. Heffernan, "Reinforcement learning for education: Opportunities and challenges," 2021, *arXiv:2107.08828*.

[15] S. Liu et al., "Reinforcement learning for clinical decision support in critical care: Comprehensive review," *J. Med. Internet Res.*, vol. 22, no. 7, 2020, Art. no. e18477.

[16] B. R. Kiran et al., "Deep reinforcement learning for autonomous driving: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4909–4926, Jun. 2022.

[17] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," 2020, *arXiv:2005.01643*.

[18] R. F. Prudencio, M. R. Maximo, and E. L. Colombini, "A survey on offline reinforcement learning: Taxonomy, review, and open problems," 2022, *arXiv:2203.01387*.

[19] E. Parisotto et al., "Stabilizing transformers for reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 7487–7498.

[20] M. Siebenborn, B. Belousov, J. Huang, and J. Peters, "How crucial is transformer in decision transformer?," 2022, *arXiv:2211.14655*.

[21] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," 2017, *arXiv:1707.03141*.

[22] K. Irie, I. Schlag, R. Csordás, and J. Schmidhuber, "Going beyond linear transformers with recurrent fast weight programmers," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 7703–7717.

[23] J. W. Rae, A. Potapenko, S. M. Jayakumar, C. Hillier, and T. P. Lillicrap, "Compressive transformers for long-range sequence modelling," in *Proc. Int. Conf. Learn. Representations*, 2020.

[24] L. Chen et al., "Decision transformer: Reinforcement learning via sequence modeling," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 15084–15097.

[25] M. Janner, Q. Li, and S. Levine, "Offline reinforcement learning as one big sequence modeling problem," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 1273–1286.

[26] P. Agarwal, A. A. Rahman, P.-L. St-Charles, S. J. Prince, and S. E. Kahou, "Transformers in reinforcement learning: A survey," 2023, *arXiv:2307.05979*.

[27] W. Li, H. Luo, Z. Lin, C. Zhang, Z. Lu, and D. Ye, "A survey on transformers in reinforcement learning," 2023, *arXiv:2301.03044*.

[28] R. Loynd, R. Fernandez, A. Celikyilmaz, A. Swaminathan, and M. Hausknecht, "Working memory graphs," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 6404–6414.

[29] A. Banino, A. P. Badia, J. Walker, T. Scholtes, J. Mitrovic, and C. Blundell, "CoBERL: Contrastive BERT for reinforcement learning," 2021, *arXiv:2107.05431*.

[30] C. Chen, Y.-F. Wu, J. Yoon, and S. Ahn, "TransDreamer: Reinforcement learning with transformer world models," 2022, *arXiv:2202.09481*.

[31] J. Sun, D.-A. Huang, B. Lu, Y.-H. Liu, B. Zhou, and A. Garg, "PlaTe: Visually-grounded planning with transformers in procedural tasks," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4924–4930, Apr. 2022.

[32] V. Micheli, E. Alonso, and F. Fleuret, "Transformers are sample efficient world models," 2022, *arXiv:2209.00588*.

[33] L. Fan et al., "MineDojo: Building open-ended embodied agents with internet-scale knowledge," 2022, *arXiv:2206.08853*.

[34] K. Paster, S. McIlraith, and J. Ba, "You can't count on luck: Why decision transformers fail in stochastic environments," 2022, *arXiv:2205.15967*.

[35] D. Brandfonbrener, A. Bietti, J. Buckman, R. Laroche, and J. Bruna, "When does return-conditioned supervised learning work for offline reinforcement learning?," 2022, *arXiv:2206.01079*.

[36] Q. Zheng, A. Zhang, and A. Grover, "Online decision transformer," 2022, *arXiv:2202.05607*.

[37] J. Shang, K. Kahatapitiya, X. Li, and M. S. Ryoo, "StARformer: Transformer with state-action-reward representations for visual reinforcement learning," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 462–479.

[38] T. Yamagata, A. Khalil, and R. Santos-Rodriguez, "Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline RL," 2022, *arXiv:2209.03993*.

[39] Y. Wang, C. Yang, Y. Wen, Y. Liu, and Y. Qiao, "Critic-guided decision transformer for offline reinforcement learning," 2023, *arXiv:2312.13716*.

[40] M. Reid, Y. Yamada, and S. S. Gu, "Can wikipedia help offline reinforcement learning?," 2022, *arXiv:2201.12122*.

[41] S. Li et al., "Pre-trained language models for interactive decision-making," 2022, *arXiv:2202.01771*.

[42] F. Liu, H. Liu, A. Grover, and P. Abbeel, "Masked autoencoding for scalable and generalizable decision making," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 12608–12618.

[43] P. Wu et al., "Masked trajectory models for prediction, representation, and control," 2023, *arXiv:2305.02968*.

[44] K.-H. Lee et al., "Multi-game decision transformers," 2022, *arXiv:2205.15241*.

[45] H. Furuta, Y. Matsuo, and S. S. Gu, "Generalized decision transformer for offline hindsight information matching," 2021, *arXiv:2111.10364*.

[46] M. Xu et al., "Prompting decision transformer for few-shot policy generalization," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 24631–24645.

[47] L. Meng et al., "Offline pre-trained multi-agent decision transformer: One big sequence model conquers all starcraftii tasks," 2021, *arXiv:2112.02845*.

[48] M. Wen et al., "Multi-agent reinforcement learning is a sequence modeling problem," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 16509–16521.

[49] S. Hu, L. Shen, Y. Zhang, and D. Tao, "Learning multi-agent communication from graph modeling perspective," in *Proc. Int. Conf. Learn. Representations*, 2024.

[50] J. Liu et al., "MaskMA: Towards zero-shot multi-agent decision making with mask-based collaborative learning," 2024, *arXiv:2310.11846*.

[51] S. Dasari and A. Gupta, "Transformers for one-shot visual imitation," in *Proc. 4th Conf. Robot Learn.*, 2021, pp. 2071–2084.

[52] A. Bucker, L. Figueredo, S. Haddadin, A. Kapoor, S. Ma, and R. Bonatti, "LATTE: Language trajectory transformer," 2022, *arXiv:2208.02918*.

[53] V. Jain, Y. Lin, E. Undersander, Y. Bisk, and A. Rai, "Transformers are adaptable task planners," 2022, *arXiv:2207.02442*.

[54] P. Ammanabrolu, E. Tien, M. Hausknecht, and M. O. Riedl, "How to avoid being eaten by a grue: Structured exploration strategies for textual worlds," 2020, *arXiv:2006.07409*.

[55] A. Adhikari et al., "Learning dynamic belief graphs to generalize on text-based games," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 3045–3057.

[56] G. Liu, A. Adhikari, A.-M. Farahmand, and P. Poupart, "Learning object-oriented dynamics for planning from text," in *Proc. Int. Conf. Learn. Representations*, 2022.

[57] W. Hao, C. Li, X. Li, L. Carin, and J. Gao, "Towards learning a generic agent for vision-and-language navigation via pre-training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13134–13143.

[58] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, "VLN BERT: A recurrent vision-and-language bert for navigation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1643–1653.

[59] S. Chen, P.-L. Guhur, C. Schmid, and I. Laptev, "History aware multimodal transformer for vision-and-language navigation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 5834–5847.

[60] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7077–7087.

[61] H. Shao, L. Wang, R. Chen, H. Li, and Y. Liu, "Safety-enhanced autonomous driving using interpretable sensor fusion transformer," 2022, *arXiv:2207.14024*.

[62] A. R. Villaflor, Z. Huang, S. Pande, J. M. Dolan, and J. Schneider, "Addressing optimism bias in sequence modeling for reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 22270–22283.

[63] F. AlMahamid and K. Grolinger, "Reinforcement learning algorithms: An overview and classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13134–13143.

[64] T. M. Moerland, J. Broekens, A. Plaat, C. M. Jonker et al., "Model-based reinforcement learning: A survey," *Found. Trends in Mach. Learn.*, vol. 16, no. 1, pp. 1–118, 2023.

[65] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, "Improving sample efficiency in model-free reinforcement learning from images," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 10674–10681.

[66] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[67] V. Mnih et al., "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

[68] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[69] T. N. Larsen, H. Ø. Teigen, T. Laache, D. Varagnolo, and A. Rasheed, "Comparing deep reinforcement learning algorithms' ability to safely navigate challenging waters," *Front. Robot. AI*, vol. 8, 2021, Art. no. 738113.

[70] K. Young and R. S. Sutton, "Understanding the pathologies of approximate policy evaluation when combined with greedification in reinforcement learning," 2020, *arXiv:2010.15268*.

[71] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[72] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2052–2062.

[73] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-learning for offline reinforcement learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 1179–1191.

[74] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit Q-learning," 2021, *arXiv:2110.06169*.

[75] A. Vaswani, "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[76] K. Han et al., "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, Jan. 2023.

[77] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Comput. Surv.*, vol. 54, pp. 200:1–200:41, 2022.

[78] Y. Liu et al., "A survey of visual transformers," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 30, 2023, doi: 10.1109/TNNLS.2022.3227717.

[79] Z. Lin et al., "A structured self-attentive sentence embedding," 2017, *arXiv:1703.03130*.

[80] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," 2016, *arXiv:1606.08415*.

[81] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," 2018, *arXiv:1803.08375*.

[82] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, pp. 99–134, 1998.

[83] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[84] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

[85] L. Espeholt et al., "IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1406–1415.

[86] S. Kapturowski, G. Ostrovski, J. Quan, R. Munos, and W. Dabney, "Recurrent experience replay in distributed reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2018.

[87] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," 2019, *arXiv:1901.02860*.

[88] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 5754–5764.

[89] T. Q. Nguyen and J. Salazar, "Transformers without tears: Improving the normalization of self-attention," 2019, *arXiv:1910.05895*.

[90] R. Xiong et al., "On layer normalization in the transformer architecture," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 10524–10533.

[91] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*.

[92] A. Van den Oord et al., "Conditional image generation with pixelcnn decoders," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 4790–4798.

[93] C. Beattie et al., "Deepmind lab," 2016, *arXiv:1612.03801*.

[94] H. F. Song et al., "V-MPO: On-policy maximum a posteriori policy optimization for discrete and continuous control," 2019, *arXiv:1909.12238*.

[95] I. Schlag, K. Irie, and J. Schmidhuber, "Linear transformers are secretly fast weight programmers," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 9355–9366.

[96] M. G. Bellemare, G. Ostrovski, A. Guez, P. Thomas, and R. Munos, "Increasing the action gap: New operators for reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 1476–1483.

[97] H. Mao et al., "Transformer in transformer as backbone for deep reinforcement learning," 2022, *arXiv:2212.14538*.

[98] A. Schwarzschild, A. Gupta, A. Ghiasi, M. Goldblum, and T. Goldstein, "The uncanny similarity of recurrence and depth," 2021, *arXiv:2102.11011*.

[99] Y. Tassa et al., "Deepmind control suite," 2018, *arXiv:1801.00690*.

[100] M. Schwarzer, A. Anand, R. Goel, R. D. Hjelm, A. Courville, and P. Bachman, "Data-efficient reinforcement learning with self-predictive representations," 2020, *arXiv:2007.05929*.

[101] W. Ye, S. Liu, T. Kurutach, P. Abbeel, and Y. Gao, "Mastering Atari games with limited data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 25476–25488.

[102] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," 2019, *arXiv:1912.01603*.

[103] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering Atari with discrete world models," 2020, *arXiv:2010.02193*.

[104] L. Kaiser et al., "Model-based reinforcement learning for Atari," 2019, *arXiv:1903.00374*.

[105] S. Ozair, Y. Li, A. Razavi, I. Antonoglou, A. Van Den Oord, and O. Vinyals, "Vector quantized models for planning," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8302–8313.

[106] R. Coulom, "Efficient selectivity and backup operators in Monte-Carlo tree search," in *Proc. Int. Conf. Comput. Games*, 2006.

[107] R. Reddy, "Speech understanding systems. summary of results of the five-year research effort at Carnegie-Mellon University," 1977.

[108] M. Carroll et al., "Towards flexible inference in sequential decision problems via bidirectional transformers," 2022, *arXiv:2204.13326*.

[109] Q. Lin, H. Liu, and B. Sengupta, "Switch trajectory transformer with distributional value approximation for multi-task reinforcement learning," 2022, *arXiv:2203.07413*.

[110] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9726–9735.

[111] R. K. Srivastava, P. Shyam, F. Mutz, W. Jaśkowski, and J. Schmidhuber, "Training agents using upside-down reinforcement learning," 2019, *arXiv:1912.02877*.

[112] A. Kumar, X. B. Peng, and S. Levine, "Reward-conditioned policies," 2019, *arXiv:1912.13465*.

[113] Y. Ma, C. Xiao, H. Liang, and H. Jianye, "Rethinking decision transformer via hierarchical reinforcement learning," 2024. [Online]. Available: https://openreview.net/forum?id=7v3tkQmtpE

[114] S. Hu, L. Shen, Y. Zhang, and D. Tao, "Graph decision transformer," 2023, *arXiv:2303.03747*.

[115] A. Ghanem, P. Ciblat, and M. Ghogho, "Multi-objective decision transformers for offline reinforcement learning," 2023, *arXiv:2308.16379*.

[116] K. Wang, H. Zhao, X. Luo, K. Ren, W. Zhang, and D. Li, "Bootstrapped transformer for offline reinforcement learning," 2022, *arXiv:2206.08569*.

[117] J. Kim, S. Lee, W. Kim, and Y. Sung, "Decision convformer: Local filtering in metaformer is sufficient for decision making," 2023, *arXiv:2310.03022*.

[118] A. Nair, M. Dalal, A. Gupta, and S. Levine, "Accelerating online reinforcement learning with offline datasets," 2020, *arXiv:2006.09359*.

[119] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," 2018, *arXiv:1805.00909*.

[120] T. Haarnoja et al., "Soft actor-critic algorithms and applications," 2018, *arXiv:1812.05905*.

[121] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4RL: Datasets for deep data-driven reinforcement learning," 2020, *arXiv:2004.07219*.

[122] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations," 2019, *arXiv:1909.11942*.

[123] Z. Liu, Z. Fan, Y. Wang, and P. S. Yu, "Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021.

[124] M. Yang, D. Schuurmans, P. Abbeel, and O. Nachum, "Dichotomy of control: Separating what you can control from what you cannot," 2022, *arXiv:2210.13435*.

[125] Y.-H. Wu, X. Wang, and M. Hamaya, "Elastic decision transformer," 2023, *arXiv:2307.02484*.

[126] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 2048–2056.

[127] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Proc. 3rd Annu. Conf. Robot Learn.*, 2020, pp. 1094–1100.

[128] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.

[129] D. Y.-T. Hui, M. Chevalier-Boisvert, D. Bahdanau, and Y. Bengio, "BabyAI 1.1," 2020, *arXiv:2007.12770*.

[130] X. Puig et al., "Virtualhome: Simulating household activities via programs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8494–8502.

[131] M. Yang and O. Nachum, "Representation matters: Offline pretraining for sequential decision making," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 11784–11794.

[132] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," 2019, *arXiv:1911.11361*.

[133] G. Cuccu, J. Togelius, and P. Cudré-Mauroux, "Playing Atari with six neurons," 2018, *arXiv:1806.01363*.

[134] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *J. Artif. Intell. Res.*, vol. 47, pp. 253–279, 2013.

[135] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[136] R. D. Shachter, "Probabilistic inference and influence diagrams," *Operations Res.*, vol. 36, no. 4, pp. 589–604, 1988.

[137] V. Sanh et al., "Multitask prompted training enables zero-shot task generalization," 2021, *arXiv:2110.08207*.

[138] J. Wei et al., "Finetuned language models are zero-shot learners," 2021, *arXiv:2109.01652*.

[139] A. Li, L. Pinto, and P. Abbeel, "Generalized hindsight for reinforcement learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 7754–7767.

[140] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[141] E. Mitchell, R. Rafailov, X. B. Peng, S. Levine, and C. Finn, "Offline meta-reinforcement learning with advantage weighting," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 7780–7791.

[142] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," 2021, *arXiv:2107.13586*.

[143] M. Shridhar et al., "ALFRED: A benchmark for interpreting grounded instructions for everyday tasks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10737–10746.

[144] S. Hu, L. Shen, Y. Zhang, and D. Tao, "Prompt-tuning decision transformer with preference ranking," 2023, *arXiv:2305.09648*.

[145] B. Trabucco, M. Phielipp, and G. Berseth, "AnyMorph: Learning transferable polices by inferring agent morphology," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 21677–21691.

[146] Y. Tang and D. Ha, "The sensory neuron as a transformer: Permutation-invariant neural networks for reinforcement learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 22574–22587.

[147] A. Boustati, H. Chockler, and D. C. McNamee, "Transfer learning with causal counterfactual reasoning in decision transformers," 2021, *arXiv:2110.14355*.

[148] Y. Yang and J. Wang, "An overview of multi-agent reinforcement learning from game theoretical perspective," 2020, *arXiv:2011.00583*.

[149] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of PPO in cooperative, multi-agent games," 2021, *arXiv:2103.01955*.

[150] M. Samvelyan et al., "The starcraft multi-agent challenge," 2019, *arXiv:1902.04043*.

[151] R. Sanjaya, J. Wang, and Y. Yang, "Measuring the non-transitivity in chess," *Algorithms*, vol. 15, no. 5, 2022, Art. no. 152.

[152] J. G. Kuba et al., "Trust region policy optimisation in multi-agent reinforcement learning," 2021, *arXiv:2109.11251*.

[153] J. G. Kuba et al., "Settling the variance of multi-agent policy gradients," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 13458–13470.

[154] C. S. de Witt, B. Peng, P.-A. Kamienny, P. Torr, W. Böhmer, and S. Whiteson, "Deep multi-agent reinforcement learning for decentralized continuous cooperative control," 2020, *arXiv:2003.06709*.

[155] K. Kurach et al., "Google research football: A novel reinforcement learning environment," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 4501–4510.

[156] Y. Yang, G. Chen, W. Wang, X. Hao, J. Hao, and P. A. Heng, "Transformer-based working memory for multiagent reinforcement learning with action parsing," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 34874–34886.

[157] S. Hu, F. Zhu, X. Chang, and X. Liang, "UPDeT: Universal multi-agent reinforcement learning via policy decoupling with transformers," 2021, *arXiv:2101.08001*.

[158] C. A. Rothkopf, D. H. Ballard, and M. M. Hayhoe, "Task and context determine where you look," *J. Vis.*, vol. 7, no. 14, pp. 1–20, 2007.

[159] R. Zhang et al., "AGIL: Learning attention from human for visuomotor tasks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 692–707.

[160] Z. Mandi, F. Liu, K. Lee, and P. Abbeel, "Towards more generalizable one-shot visual imitation learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 2434–2444.

[161] A. Bucker, L. Figueredo, S. Haddadin, A. Kapoor, S. Ma, and R. Bonatti, "Reshaping robot trajectories using natural language commands: A study of multi-modal data alignment using transformers," 2022, *arXiv:2203.13411*.

[162] A. Radford et al., "Learning transferable visual models from natural language supervision," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8748–8763.

[163] P. Sharma et al., "Correcting robot plans with natural language feedback," 2022, *arXiv:2204.05186*.

[164] P.-L. Guhur, S. Chen, R. Garcia, M. Tapaswi, I. Laptev, and C. Schmid, "Instruction-driven history-aware policies for robotic manipulations," 2022, *arXiv:2209.04899*.

[165] Y. Han et al., "Learning generalizable vision-tactile robotic grasping strategy for deformable objects via transformer," 2021, *arXiv:2112.06374*.

[166] R. Jangir, N. Hansen, S. Ghosal, M. Jain, and X. Wang, "Look closer: Bridging egocentric and third-person views with transformers for robotic manipulation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 3046–3053, Apr. 2022.

[167] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang, "Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers," 2021, *arXiv:2107.03996*.

[168] H. Kim, Y. Ohmura, and Y. Kuniyoshi, "Transformer-based deep imitation learning for dual-arm robot manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 8965–8972.

[169] M.-A. Côté et al., "TextWorld: A learning environment for text-based games," in *Proc. Workshop Comput. Games*, 2018, pp. 41–75.

[170] J. Urbanek et al., "Learning to speak and act in a fantasy text adventure game," 2019, *arXiv:1903.03094*.

[171] M. Hausknecht, P. Ammanabrolu, M.-A. Côté, and X. Yuan, "Interactive fiction games: A colossal adventure," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 7903–7910.

[172] X. Yuan et al., "Interactive language learning by question answering," 2019, *arXiv:1908.10909*.

[173] P. Ammanabrolu and M. O. Riedl, "Playing text-adventure games with graph-based deep reinforcement learning," 2018, *arXiv:1812.01628*.

[174] P. Ammanabrolu and M. Hausknecht, "Graph constrained reinforcement learning for natural language action spaces," 2020, *arXiv:2001.08837*.

[175] M. Tuli, A. C. Li, P. Vaezipoor, T. Q. Klassen, S. Sanner, and S. A. McIlraith, "Learning to follow instructions in text-based games," 2022, *arXiv:2211.04591*.

[176] P. Anderson et al., "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3674–3683.

[177] F. Taioli et al., "Mind the error! detection and localization of instruction errors in vision-and-language navigation," 2024, *arXiv:2403.10700*.

[178] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi, "TOUCHDOWN: Natural language navigation and spatial reasoning in visual street environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12538–12547.

[179] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer, "Vision-and-dialog navigation," in *Proc. 3rd Annu. Conf. Robot Learn.*, 2020, pp. 394–406.

[180] K. Nguyen, D. Dey, C. Brockett, and B. Dolan, "Vision-based navigation with language-based assistance via imitation learning with indirect intervention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12527–12537.

[181] K. Nguyen and H. Daumé III, "Help, Anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning," 2019, *arXiv:1909.01871*.

[182] Y. Qi et al., "REVERIE: Remote embodied visual referring expression in real indoor environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9979–9988.

[183] D. Misra, J. Langford, and Y. Artzi, "Mapping instructions and visual observations to actions with reinforcement learning," 2017, *arXiv:1704.08795*.

[184] H. Tan and M. Bansal, "LXMERT: Learning cross-modality encoder representations from transformers," 2019, *arXiv:1908.07490*.

[185] X. Li et al., "Robust navigation with language pretraining and stochastic sampling," 2019, *arXiv:1909.02244*.

[186] A. Majumdar, A. Shrivastava, S. Lee, P. Anderson, D. Parikh, and D. Batra, "Improving vision-and-language navigation with image-text pairs from the web," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 259–274.

[187] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 13–23.

[188] D. An, Y. Qi, Y. Li, Y. Huang, L. Wang, T. Tan, and J. Shao, "BEVBert: Multimodal map pre-training for language-guided navigation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 2737–2748.

[189] S. Paul, A. K. Roy-Chowdhury, and A. Cherian, "AVLEN: Audio-visual-language embodied navigation in 3D environments," 2022, *arXiv:2210.07940*.

[190] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev, "Think global, act local: Dual-scale graph transformer for vision-and-language navigation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16516–16526.

[191] A. Pashevich, C. Schmid, and C. Sun, "Episodic transformer for vision-and-language navigation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 15922–15932.

[192] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, 1–16.

[193] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9328–9337.

[194] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11618–11628.

[195] A. Zhao, T. He, Y. Liang, H. Huang, G. Van den Broeck, and S. Soatto, "SAM: Squeeze-and-mimic networks for conditional visual driving policy learning," in *Proc. 1st Annu. Conf. Robot Learn.*, 2021, pp. 156–175.

[196] S. Hu, L. Chen, P. Wu, H. Li, J. Yan, and D. Tao, "ST-P3: End-to-end vision-based autonomous driving via spatial-temporal feature learning," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 533–549.

[197] Y. Hu et al., "Planning-oriented autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 17853–17862.

[198] N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto, "Behavior transformers: Cloning k modes with one stone," 2022, *arXiv:2206.11251*.

[199] S. Li et al., "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 5244–5254.

[200] X. Lin, L. Yu, K.-T. Cheng, and Z. Yan, "BATFormer: Towards boundary-aware lightweight transformer for efficient medical image segmentation," *IEEE J. Biomed. Health Inform.*, vol. 27, no. 7, pp. 3501–3512, Jul. 2023.

[201] E. Parisotto and R. Salakhutdinov, "Efficient transformers in reinforcement learning using actor-learner distillation," 2021, *arXiv:2104.01655*.

[202] M. Laskin et al., "In-context reinforcement learning with algorithm distillation," 2022, *arXiv:2210.14215*.

[203] I. Schubert et al., "A generalist dynamics model for control," 2023, *arXiv:2305.10912*.

[204] C. Snell, M. Yang, J. Fu, Y. Su, and S. Levine, "Context-aware language modeling for goal-oriented dialogue systems," 2022, *arXiv:2204.10198*.

[205] S. Verma, J. Fu, M. Yang, and S. Levine, "CHAI: A chatbot AI for task-oriented dialogue with offline reinforcement learning," 2022, *arXiv:2204.08426*.

[206] L. Ouyang et al., "Training language models to follow instructions with human feedback," 2022, *arXiv:2203.02155*.

Shengchao Hu received the BE degree in computer science from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 2022. He is currently working towards the PhD degree in computer science with SJTU. His research interests include autonomous driving, reinforcement learning, and machine learning.

Li Shen received the bachelor's and PhD degrees from the School of Mathematics, South China University of Technology. He is currently an associate professor with Sun Yat-sen University. Previously, he was a research scientist with JD Explore Academy, Beijing, and a senior researcher with Tencent AI Lab, Shenzhen. His research interests include theory and algorithms for nonsmooth convex and nonconvex optimization, and their applications in trustworthy artificial intelligence, deep learning, and reinforcement learning.

Ya Zhang received the bachelor's degree from Tsinghua University, and the PhD degree in information sciences and technology from Pennsylvania State University. She is currently a professor with Cooperative Medianet Innovation Center, Shanghai Jiao Tong University. Her research focuses on machine learning applications in multimedia and healthcare. She has published more than 200 scholarly articles in high-impact international journals and conferences, with her work being cited more than 10 K times according to Google Scholar. Her achievements are recognized through various awards and honors, including earning a Leading Talent under the National High-Level Talent Special Support Program (2023), and receiving the First Prize of the Shanghai Technological Invention Award (2022).

Yixin Chen (Fellow, IEEE) is a professor in computer science and engineering with Washington University, St Louis. His research interests include data mining, machine learning, artificial intelligence, and optimization. He has also served as a program chair for IEEE International Conference on Big Data (2021), and an associate editor for *ACM Transactions on Computing for Healthcare*, *ACM Transactions of Intelligent Systems and Technology*, *Annals of Mathematics and Artificial Intelligence*, *Journal of Artificial Intelligence Research*, and *IEEE Transactions on Knowledge and Data Engineering*.

Dacheng Tao (Fellow, IEEE) is currently a distinguished university professor with the College of Computing & Data Science, Nanyang Technological University. He mainly applies statistics and mathematics to artificial intelligence and data science, and his research is detailed in one monograph and more than 200 publications in prestigious journals and proceedings at leading conferences, with best paper awards, best student paper awards, and test-of-time awards. His publications have been cited more than 120 K times and he has an h-index 170+ in Google Scholar. He received the 2015 and 2020 Australian Eureka Prize, the 2018 IEEE ICDM Research Contributions Award, and the 2021 IEEE Computer Society McCluskey Technical Achievement Award. He is a fellow of the Australian Academy of Science, AAAS, and ACM.